SVEUČILIŠTE U ZAGREBU
PRIRODOSLOVNO–MATEMATIČKI FAKULTET
MATEMATIČKI ODSJEK

Sara Markić

PRIMJENA STATISTIČKIH METODA U OPTIMIZACIJI MARKETINŠKIH AKTIVNOSTI U MALOPRODAJI

Diplomski rad

Voditelj rada:
prof. dr. sc. Siniša Slijepčević

Zagreb, studeni 2017.

Ovaj diplomski rad obranjen je dana _____ pred ispitnim povjerenstvom u sastavu:

1. _____, predsjednik
2. _____, član
3. _____, član

Povjerenstvo je rad ocijenilo ocjenom _____.

Potpisi članova povjerenstva:

1. _____
2. _____
3. _____

Hvala svima koji su bili uz mene i učinili sve teškoće podnošljivima. Vaši načini bili su različiti, ali ishod je bio isti - pomogli ste mi da dođem do ovog trenutka. Osobito hvala mojim roditeljima na potpori i ljubavi te Vjeranu što je izdržao sav ovaj kaos i vrtuljak emocija koji je nastao upisom ovog fakulteta – obećajem, neću više upisivati ništa slično.

Sadržaj

Popis slika

Popis algoritama

Popis tablica

Uvod

U ovom diplomskom radu promatra se problem predviđanja ishoda marketinške aktivnosti. Promatrana marketinška aktivnost u kontekstu ovog rada je promocija proizvoda koja nakon nekog vremena trajanja kao ishod ima kontinuiranu varijablu – ona može predstavljati vrijednost koja sažima profit, vidljivost ili druge mjerljive utjecaje promocije. Modelira se odnos karakteristika promocije i ciljne varijable. Naučeni model moguće je iskoristiti za dobru procjenu ishoda marketinške aktivnosti za dane karakteristike promocije te se time olakšava pronalaženje promocije s najboljim ishodom.

Učenje modela vrši se metodama razvijenim u području matematičke statistike i strojnog učenja. Za predviđanje se razmatraju linearni i nelinearni statistički modeli. Obrađeni su modeli linearne regresije i višestruka aditivna regresijska stabla. Naučen model za kojeg se dobro zna njegova pouzdanost može biti od velike koristi poduzeću za istraživanje i provjeru ishoda marketinške aktivnosti.

U poglavlju 1 dana je teoretska analiza linearne regresije kroz okvir linearne algebre. Dotaknuti su pristupi računanja sustava jednadžbi koji proizlaze iz samog problema. Model linearne regresije opisan je u kontekstu strojnog učenja.
U poglavlju 2 dan je pregled za regresijska stabla i postupke učenja skupine stabala. Opisani su standardni algoritam gradijentnog *boosting*-a i njegova primjena u kontekstu stabala odluke. Opisana je i nadogradnja koja je korištena za vrednovanje.
U poglavlju 3 prisutan je detaljan grafički i tekstualan opis podataka te je analizirana interakcija varijabli s varijablom koju se predviđa.
U poglavlju 4 opisan je postupak vrednovanja modela, navedeni su korišteni alati i statističke metode, dana je statistička usporedba između modela i osvrt na praktičnu korist najboljeg pronađenog modela.

Poglavlje 1

Linearna regresija

Tijekom razmatranja podataka dobivenih znanstvenim istraživanjem često se postavlja pitanje postoji li uzročna veza između dvije ili više varijabli. Proces pronalaženja matematičke formule koja najbolje opisuje šumovite podatke se zove regresijska analiza. Najraniji oblik linearne regresije je metoda najmanjih kvadrata koja je prvi put objavljena početkom 19. stoljeća.

Cilj regresijske analize je pronaći vezu između varijabli kako bi se mogao konstruirati model koji se može koristiti u svrhu predviđanja. Slučajna varijabla koja se predviđa se označava sa Y i još se naziva zavisnom varijablom ili varijablom odgovora, a nezavisne varijable korištene za predviđanje Y se označavaju sa x_i i još se zovu regresorske varijable.

Postoje tri tipa regresije: jednostavna linearna regresija, višestruka linearna regresija i nelinearna regresija. U ovom radu koristit će se višestruka linearna regresija koja opisuje linearnu vezu između jedne zavisne i više nezavisnih varijabli, no, za početak ćemo se upoznati s jednostrukom. Pregled teoretskog dijela temeljen je na sadržaju izvora [8, 16].

1.1 Jednostruka linearna regresija

Jednostavna ili jednostruka linearna regresija opisuje linearnu vezu između jedne zavisne varijable i jedne nezavisne varijable. Jednostavni regresijski model je oblika:

$$y = \beta_0 + \beta_1 x + \varepsilon \tag{1.1}$$

pri čemu ε predstavlja slučajnu varijablu, tj. slučajnu grešku, a slobodni član β_0 i koeficijent smjera β_1 su nepoznati regresijski parametri.

Pretpostavke linearne regresije su da je veza između zavisne varijable i odabranog skupa nezavisnih varijabli linearna te da vrijede Gauss-Markovljevi uvjeti.

Definicija 1.1.1. Za slučajne greške ε_i, $i = 1, 2, \ldots, n$ Gauss-Markovljevi uvjeti su:

1. ε je n-dimenzionalna slučajna varijabla za koju vrijedi da je normalno distribuirana s očekivanjem $\mathrm{E}(\varepsilon_i) = 0$ i konstantnom varijancom $\mathrm{Var}(\varepsilon_i) = \sigma^2, i = 1, \ldots, n$.

2. Slučajne greške $\varepsilon \sim N(0, \sigma^2 I)$ su nezavisne jednako distribuirane slučajne varijable: $\mathrm{Cov}(\varepsilon_i, \varepsilon_j) = \mathrm{E}(\varepsilon_i \varepsilon_j) = 0, i \neq j, i, j = 1, \ldots, n$

Metoda najmanjih kvadrata

Motivacija iza metode najmanjih kvadrata kod jednostruke linearne regresije je pronaći procjene parametara tako da se odabere regresijski pravac koji je "najbliži" svim podacima. Cilj je dakle pronaći procjenitelje b_0 i b_1 takve da je suma kvadrata razlike između stvarne varijable odgovora y_i i predviđene varijable odgovora $\hat{y}_i = \beta_0 + \beta_1 x_i$ minimalna s obzirom na sve moguće vrijednosti regresijskih koeficijenata β_0 i β_1, to jest:

$$(b_0, b_1) = \underset{(\beta_0, \beta_1)}{\operatorname{argmin}} \sum_{i=1}^{n} \left[y_i - (\beta_0 + \beta_1 x_i) \right]^2. \tag{1.2}$$

Procjenitelji se dobiju rješavanjem sljedećeg sustava jednadžbi:

$$\frac{\partial}{\partial \beta_0} \sum_{i=1}^{n} \left[y_i - (\beta_0 + \beta_1 x_i) \right]^2 = 0 \tag{1.3}$$

$$\frac{\partial}{\partial \beta_1} \sum_{i=1}^{n} \left[y_i - (\beta_0 + \beta_1 x_i) \right]^2 = 0 \tag{1.4}$$

Pretpostavimo da su b_0 i b_1 rješenja gornjeg sustava jednadžbi. Tada možemo opisati vezu između x i y s regresijskim pravcem $\hat{y} = b_0 + b_1 x$ koji najbolje opisuje podatke. Nužan uvjet za lokalni ekstrem nakon deriviranja 1.3 i 1.4 glasi:

$$2 \sum_{i=1}^{n} (b_0 + b_1 x_i - y_i) = 0 \tag{1.5}$$

$$2 \sum_{i=1}^{n} (b_0 + b_1 x_i - y_i) x_i = 0 \tag{1.6}$$

Rješavamo dalje:

$$b_0 n + b_1 \sum_{i=1}^{n} x_i - \sum_{i=1}^{n} y_i = 0 \tag{1.7}$$

$$b_0 \sum_{i=1}^{n} x_i + b_1 \sum_{i=1}^{n} x_i^2 - \sum_{i=1}^{n} x_i y_i = 0 \tag{1.8}$$

Uvodimo sljedeće oznake: $\overline{x} = \frac{\sum_{i=1}^{n} x_i}{n}$, $S_{XX} = \sum_{i=1}^{n} (x_i - \overline{x})^2$ i $S_{XY} = \sum_{i=1}^{n} (x_i - \overline{x})(y_i - \overline{y})$.

$$b_0 n + b_1 n\overline{x} - n\overline{y} = 0 \quad / \cdot \overline{x} \tag{1.9}$$

$$b_0 n\overline{x} + b_1 (S_{XX} + n\overline{x}^2) - (S_{XY} + n\overline{x}\overline{y}) = 0 \tag{1.10}$$

Iz 1.9 i 1.10 dobije se $b_1 = \frac{S_{XY}}{S_{XX}}$ i $b_0 = \overline{y} - b_1\overline{x}$.
Definiramo funkciju $L(\beta_0, \beta_1) := \sum_{i=1}^{n} (y_i - (\beta_0 + \beta_1 x_i))^2$ i provjeravamo dovoljan uvjet za lokalni ekstrem pomoću Hesseove matrice:

$$H = \begin{bmatrix} \partial_\alpha^2 L(b_0, b_1) & \partial_\alpha \partial_\beta L(b_0, b_1) \\ \partial_\beta \partial_\alpha L(b_0, b_1) & \partial_\beta^2 L(b_0, b_1) \end{bmatrix} = \begin{bmatrix} 2n & 2\sum_{i=1}^{n} x_i \\ 2\sum_{i=1}^{n} x_i & 2\sum_{i=1}^{n} x_i^2 \end{bmatrix}. \tag{1.11}$$

Vrijedi $n > 0$ i

$$\begin{vmatrix} 2n & 2\sum_{i=1}^{n} x_i \\ 2\sum_{i=1}^{n} x_i & 2\sum_{i=1}^{n} x_i^2 \end{vmatrix} = 4n \sum_{i=1}^{n} x_i^2 - 4\left(\sum_{i=1}^{n} x_i\right)^2 > 0. \tag{1.12}$$

zbog aritmetičko-kvadratne nejednakosti ($K_n(x) \geqslant A_n(x)$, gdje je $A_n(x) = \frac{x_1 + x_2 + \cdots + x_n}{n}$, a $K_n(x) = \sqrt{\frac{x_1^2 + x_2^2 + \cdots + x_n^2}{n}}$) i pretpostavke s početka. Hesseova matrica je pozitivno definitna te po [10, teoremu 16.7] o dovoljnim uvjetima za lokalni ekstrem funkcija L ima u točki (b_0, b_1) strogi lokalni minimum. Budući da je stacionarna točka jedinstvena, taj lokalni minimum je ujedno i jedinstveni globalni minimum.

Procijenjena vrijednost regresije je definirana kao $\hat{y}_i = b_0 + b_1 x_i$. Razlika između y_i i procijenjene vrijednosti $\hat{y}_i$, $e_i = y_i - \hat{y}_i$ se zove regresijski rezidual. Rezidual je za razliku od slučajnih grešaka ili šumova ε_i izmjerljiv.

Svojstva procjenitelja b_0 i b_1

U ovom poglavlju pokazat će se statistička svojstva procjenitelja za jednostruku linearnu regresiju. Za početak se neće uzimati pretpostavke o distribuciji slučajnih grešaka, ali pretpostavit će se $\mathrm{E}(\varepsilon_i) = 0$, $\mathrm{Var}(\varepsilon_i) = \sigma^2$ i da su ε_i za $i = 1, 2, \ldots, n$ nezavisni.

Definicija 1.1.2. Procjenitelj $T_n = f(X_1, ..., X_n)$ je nepristrani procjenitelj za parametar τ ako vrijedi $\mathrm{E}(T_n) = \tau$.

Teorem 1.1.1. Procjenitelj dobiven metodom najmanjih kvadrata b_0 je nepristrani procjenitelj od β_0.

Dokaz.

$$\begin{aligned}
\mathrm{E}(b_0) &= \mathrm{E}\left(\overline{y} - b_1\overline{x}\right) \\
&= \mathrm{E}\left(\frac{1}{n}\sum_{i=1}^{n} y_i\right) - \mathrm{E}(b_1)\overline{x} \\
&= \frac{1}{n}\sum_{i=1}^{n}\mathrm{E}(y_i) - \overline{x}\,\mathrm{E}(b_1) \\
&= \frac{1}{n}\sum_{i=1}^{n}(\beta_0 + \beta_1 x_i) - \beta_1\overline{x} \\
&= \frac{1}{n}\sum_{i=1}^{n}\beta_0 + \beta_1\frac{1}{n}\sum_{i=1}^{n} x_i - \beta_1\overline{x} \\
&= \beta_0.
\end{aligned}$$

□

Teorem 1.1.2. Procjenitelj dobiven metodom najmanjih kvadrata b_1 je nepristrani procjenitelj od β_1.

Dokaz.

$$\begin{aligned}
\mathrm{E}(b_1) &= \mathrm{E}\left(\frac{S_{xy}}{S_{xx}}\right) \\
&= \frac{1}{S_{xx}}\,\mathrm{E}\left(\frac{1}{n}\sum_{i=1}^{n}(y_i - \overline{y})(x_i - \overline{x})\right) \\
&= \frac{1}{S_{xx}}\frac{1}{n}\sum_{i=1}^{n}(x_i - \overline{x})\,\mathrm{E}(y_i) \\
&= \frac{1}{S_{xx}}\frac{1}{n}\sum_{i=1}^{n}(x_i - \overline{x})(\beta_0 + \beta_1 x_i) \\
&= \frac{1}{S_{xx}}\frac{1}{n}\sum_{i=1}^{n}(x_i - \overline{x})\beta_1 x_i \\
&= \frac{1}{S_{xx}}\frac{1}{n}\sum_{i=1}^{n}(x_i - \overline{x})\beta_1(x_i - \overline{x}) \\
&= \frac{1}{S_{xx}}\frac{1}{n}\sum_{i=1}^{n}(x_i - \overline{x})^2\beta_1 = \frac{S_{xx}}{S_{xx}}\beta_1 = \beta_1.
\end{aligned}$$

□

Teorem 1.1.3. $\mathrm{Var}(b_1) = \frac{\sigma^2}{nS_{xx}}$.

Dokaz.

$$\begin{aligned}
\operatorname{Var}(b_1) &= \operatorname{Var}\left(\frac{S_{xy}}{S_{xx}}\right) \\
&= \frac{1}{S_{xx}^2} \operatorname{Var}\left(\frac{1}{n}\sum_{i=1}^{n}(y_i - \overline{y})(x_i - \overline{x})\right) \\
&= \frac{1}{S_{xx}^2} \operatorname{Var}\left(\frac{1}{n}\sum_{i=1}^{n} y_i(x_i - \overline{x})\right) \\
&= \frac{1}{S_{xx}^2}\frac{1}{n^2}\sum_{i=1}^{n}(x_i - \overline{x})^2 \operatorname{Var}(y_i) \\
&= \frac{1}{S_{xx}^2}\frac{1}{n^2}\sum_{i=1}^{n}(x_i - \overline{x})^2\sigma^2 \\
&= \frac{\sigma^2}{nS_{xx}}.
\end{aligned}$$

□

Teorem 1.1.4. Procjenitelj dobiven metodom najmanjih kvadrata b_1 i $\overline{y}$ su nekorelirani. Uz pretpostavku o normalnosti y_i za $i = 1, 2, \ldots, n$, b_1 i $\overline{y}$ su normalno distribuirani i nezavisni.

Dokaz.

$$\begin{aligned}
\operatorname{Cov}(b_1, \overline{y}) &= \operatorname{Cov}\left(\frac{S_{xy}}{S_{xx}}, \overline{y}\right) \\
&= \frac{1}{S_{xx}} \operatorname{Cov}(S_{xx}, \overline{y}) \\
&= \frac{1}{nS_{xx}} \operatorname{Cov}\left(\sum_{i=1}^{n}(x_i - \overline{x})(y_i - \overline{y}), \overline{y}\right) \\
&= \frac{1}{nS_{xx}} \operatorname{Cov}\left(\sum_{i=1}^{n}(x_i - \overline{x})y_i, \overline{y}\right) \\
&= \frac{1}{n^2S_{xx}} \operatorname{Cov}\left(\sum_{i=1}^{n}(x_i - \overline{x})y_i, \sum_{i=1}^{n} y_i\right) \\
&= \frac{1}{n^2S_{xx}} \sum_{i,j=1}^{n}(x_i - \overline{x}) \operatorname{Cov}(y_i, y_j)
\end{aligned}$$

S obzirom na to da vrijedi $\mathrm{E}(\varepsilon_i) = 0$ i da su ε_i nezavisni, možemo pisati:

$$\mathrm{Cov}(y_i, y_j) = \mathrm{E}[(y_i - \mathrm{E}(y_i))(y_j - \mathrm{E}(y_j))] = \mathrm{E}(\varepsilon_i, \varepsilon_j) = \begin{cases} \sigma^2, \text{ako } i = j \\ 0, \text{ako } i \neq j \end{cases}$$

Dakle, zaključujemo da

$$\mathrm{Cov}(b_1, \overline{y}) = \frac{1}{n^2 S_{xx}} \sum_{i=1}^{n} (x_i - \overline{x})\sigma^2 = 0.$$

Prisjetimo se da ako je korelacija jednaka nuli, to je ekvivalentno nezavisnosti između dvije normalne varijable. Zato zaključujemo da su b_0 i $\overline{y}$ nezavisni. □

Teorem 1.1.5. $\mathrm{Var}(b_0) = \left(\frac{1}{n} + \frac{\overline{x}^2}{nS_{xx}}\right)\sigma^2$.

Dokaz.

$$\begin{aligned} \mathrm{Var}(b_0) &= \mathrm{Var}(\overline{y} - b_1\overline{x}) \\ &= \mathrm{Var}(\overline{y}) + (\overline{x})^2 \mathrm{Var}(b_1) \\ &= \frac{\sigma^2}{n} + \overline{x}^2 \frac{\sigma^2}{nS_{xx}} \\ &= (\frac{1}{n} + \frac{\overline{x}^2}{nS_{xx}})\sigma^2. \end{aligned}$$

□

Svojstva teorema 1.1.1. - 1.1.5., a posebno varijance od b_0 i b_1 su važna kad želimo vršiti statističko zaključivanje o nagibu i pomaku jednostavne linearne regresije.

S obzirom da varijance od b_0 i b_1 sadržavaju varijancu slučajne greške koja nam je nepoznata, nju trebamo procijeniti. Neka je y_i opažena varijabla odgovora i $\hat{y}_i = b_0 + b_1 x_i$ procijenjena vrijednost odgovora. Poznati su nam y_i i $\hat{y}_i$, dok je prava greška modela σ_i nepoznata i želimo je procijeniti. Empirijska verzija greške ε_i je vrijednost $y_i - \hat{y}_i$, tj. regresijski rezidual. Predlaže se sljedeća procjena varijance pogreške:

$$s^2 = \frac{1}{n-2} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \tag{1.13}$$

s^2 je nepristrani procjenitelj varijance pogreška σ^2 jer se može pokazati da za linearni regresijski model s p parametara nazivnik u 1.13 treba biti $n - p$ da bi procjenitelj bio nepristran. [16, str. 16-18]

Metoda maksimalne vjerodostojnosti

Ako pretpostavimo da je zavisna varijabla y_i normalno distribuirana: $y_i \sim N(\beta_0 + \beta_1 x_i, \sigma^2)$, možemo definirati funkciju vjerodostojnosti za $(y_1, y_2, \ldots, y_n)$:

$$L = \prod_{i=1}^{n} f(y_i) = \frac{1}{(2\pi)^{n/2}\sigma^n} e^{(-1/2\sigma^2)\sum_{i=1}^{n}(y_i-\beta_0-\beta_1 x_i)^2} \tag{1.14}$$

.

Procjenitelji od β_0 i β_1 koji maksimiziraju funkciju vjerodostojnosti L su ekvivalentni procjeniteljima koji minimiziraju eksponencijalni dio te funkcije, koji su pak jednaki procjeniteljima dobivenim metodom najmanjih kvadrata.

Nakon što dobijemo b_1 i b_0, možemo izračunati procijenjenu vrijednost $\hat{y}_i$ i funkciju vjerodostojnosti u terminima procijenjenih vrijednosti:

$$L = \prod_{i=1}^{n} f(y_i) = \frac{1}{(2\pi)^{n/2}\sigma^n} e^{(-1/2\sigma^2)\sum_{i=1}^{n}(y_i-\hat{y}_i)^2}$$

Uzmemo parcijalnu derivaciju po σ^2 u funkciji $\log(L)$ i izjednačimo s nulom:

$$\frac{\partial \log(L)}{\partial \sigma^2} = -\frac{n}{2\sigma^2} + \frac{1}{2\sigma^4}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 = 0$$

Procjenitelj maksimalne vjerodostojnosti od σ^2 je $\hat{\sigma^2} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$. Primjećujemo da je $\hat{\sigma}^2$ pristrani procjenitelj, dok je $\frac{n}{n-2}\hat{\sigma}^2$ nepristrani procjenitelj od σ^2.

1.2 Višestruka linearna regresija

Višestruka linearna regresija je model s jednom zavisnom i više nezavisnih varijabli koji pretpostavlja da je varijabla odgovora linearna funkcija parametara modela. Opći oblik modela višestruke linearne regresije je dan sa:

$$y_i = \beta_0 + \beta_1 x_{1i} + \beta_2 x_{2i} + \cdots + \beta_k x_{ki} + \varepsilon_i \tag{1.15}$$

gdje je y_i zavisna varijabla, $\beta_0, \beta_1, \ldots, \beta_k$ su regresijski koeficijenti, a ε_i su slučajne greške uz pretpostavku da je $\mathrm{E}(\varepsilon_i) = 0$ i $\mathrm{Var}(\varepsilon_i) = \sigma^2$ za $i = 1, 2, \ldots, n$. U klasičnim regresijskim uvjetima pretpostavlja se da su slučajne greške normalno distribuirane s konstantnom varijancom σ^2. Regresijski koeficijenti se procjenjuju koristeći metodu najmanjih kvadrata i tada pretpostavka o normalnosti nije nužna. No, uz tu pretpostavku, procjenitelji dobiveni metodom najmanjih kvadrata su isti kao i procjenitelji dobiveni metodom maksimalne vjerodostojnosti.

Kako bismo pronašli regresijske koeficijente, na osnovi uzorka veličine n rješavamo jednadžbu 1.15 zapisanu kao sustav jednadžbi:

$$y = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon} \tag{1.16}$$

gdje je:

$$\boldsymbol{X} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1k} \\ x_{21} & x_{22} & \cdots & x_{2k} \\ \cdots & & & \\ x_{n1} & x_{n2} & \cdots & x_{nk} \end{bmatrix} \quad \boldsymbol{\beta} = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \cdots \\ \beta_{k-1} \end{bmatrix} \quad \boldsymbol{\varepsilon} = \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \cdots \\ \varepsilon_n \end{bmatrix} \quad \boldsymbol{y} = \begin{bmatrix} y_1 \\ y_2 \\ \cdots \\ y_n \end{bmatrix}$$

Procjena od $\boldsymbol{\beta}$ metodom najmanjih kvadrata može se dobiti tako da se minimizira suma kvadrata rezidualnih odstupanja:

$$b = \underset{\beta}{\operatorname{argmin}}[(y - X\beta)^\tau (y - X\beta)] \tag{1.17}$$

gdje je $\boldsymbol{b^\tau} = (b_0, b_1, \ldots, b_{k-1})^\tau$ k-dimenzionalni vektor procjena regresijskih koeficijenata, a $(y - X\beta)^\tau (y - X\beta)$ se naziva srednjom kvadratnom pogreškom (engl. *MSE, mean squared error*).
Dalje u tekstu će se radi jednostavnosti umjesto oznaka matrica i vektora $\boldsymbol{X}$, $\boldsymbol{\beta}$, $\boldsymbol{\varepsilon}$ i $\boldsymbol{y}$ koristiti oznake X, β, ε i y.
Jednadžba koju zovemo normalnom jednadžbom je:

$$b = (X^\tau X)^{-1} X^\tau y \tag{1.18}$$

Može se pokazati da je normalna jednadžba konveksna funkcija.

Definicija 1.2.1. Neka je X konveksni skup u realnom vektorskom prostoru i neka je $f : X \to \mathbb{R}$. Kažemo da je f konveksna funkcija ako vrijedi:

$$\forall x_1, x_2 \in X, \forall t \in [0,1] \quad f(tx_1 + (1-t)x_2) \leqslant tf(x_1) + (1-t)f(x_2).$$

Želimo pokazati da je

$$J : \beta \mapsto \|y - X\beta\|^2 = \|y\|^2 - 2y^\tau X\beta + \beta^\tau X^\tau X\beta$$

konveksna funkcija. Po definiciji 1.2.1, dovoljno je pokazati da:

$$J(t\beta_1 + (1-t)\beta_2) - [tJ(\beta_1) + (1-t)J(\beta_2)] \leqslant 0$$

za sve β_1 i β_2 iz vektorskog prostora i za sve $t \in [0,1]$. Nakon računanja, lijeva strana postaje:

$$t^2\beta_1^\tau X^\tau\beta_1 + (1-t)^2\beta_2^\tau X^\tau X\beta_2 + 2t(1-t)\beta_1^\tau X^\tau X\beta_2 - t\beta_1^\tau X^\tau X\beta_1 - (1-t)\beta_2^\tau X^\tau X\beta_2$$
$$= -t(1-t)[(\beta_1-\beta_2)^\tau X^\tau X(\beta_1-\beta_2)] = -t(1-t)\|X(\beta_1-\beta_2)\|^2$$

što je očito $\leqslant 0$, pa smo time dokazali da je funkcija kvadratne pogreške konveksna.

Teorem 1.2.1. Procjena od β metodom najmanjih kvadrata za model višestruke linearne regresije $y = X\beta + \varepsilon$ je $b = (X^\tau X)^{-1}X^\tau y$ uz pretpostavku da je $X^\tau X$ nesingularna matrica. To je ekvivalentno pretpostavci da su vektori stupci od X nezavisni.

Dokaz. Da dobijemo procjenu od β metodom najmanjih kvadrata trebamo minimizirati rezidual sume kvadrata tako da riješimo sljedeću jednadžbu:

$$\frac{\partial}{\partial b}[(y-Xb)^\tau(y-Xb)] = 0, \tag{1.19}$$

ili ekvivalentno:

$$\frac{\partial}{\partial b}[(y^\tau y - 2y^\tau Xb + b^\tau X^\tau Xb)] = 0. \tag{1.20}$$

Uzmemo li parcijalnu derivaciju s obzirom na svaku komponentu od β dobijemo sljedeću normalnu jednadžbu:

$$X^\tau Xb = X^\tau y. \tag{1.21}$$

S obzirom na to da $X^\tau X$ nije singularna, slijedi da je $b = (X^\tau X)^{-1}X^\tau y$ čime je dokaz gotov. □

Teorem 1.2.2. Procjenitelj $b = (X^\tau X)^{-1}X^\tau y$ je nepristrani procjenitelj od β. Nadalje,

$$\mathrm{Var}(b) = (X^\tau X)^{-1}\sigma^2. \tag{1.22}$$

Dokaz. Primjećujemo da:

$$\mathrm{E}(b) = \mathrm{E}((X^\tau X)^{-1}X^\tau y) = (X^\tau X)^{-1}X^\tau\,\mathrm{E}(y) = (X^\tau X)^{-1}X^\tau X\beta = \beta. \tag{1.23}$$

čime je pokazana nepristranost od b. Varijanca se može direktno izračunati:

$$\mathrm{Var}(b) = \mathrm{Var}((X^\tau X)^{-1}X^\tau y) = (X^\tau X)^{-1}X^\tau\,\mathrm{Var}(b)((X^\tau X)^{-1}X^\tau)^\tau$$
$$= (X^\tau X)^{-1}X^\tau X(X^\tau X)^{-1}\sigma^2 = (X^\tau X)^{-1}\sigma^2.$$

□

Pošto je σ^2 nemjerljiva veličina, njena procjena će utjecati na zaključivanje o regresijskim koeficijentima. Kako bismo procijenili σ^2 prvo promatramo rezidual sume kvadrata:

$$e^\tau e = (y - Xb)^\tau (y - Xb) = y^\tau [I - X(X^\tau X)^{-1} X^\tau] y = y^\tau P y. \tag{1.24}$$

To je zapravo mjera udaljenosti između opaženog y i procijenjene vrijednosti $\hat{y}$. Lako se provjeri da je $P = [I - X(X^\tau X)^{-1} X^\tau]$ idempotentna matrica:

$$P^2 = [I - X(X^\tau X)^{-1} X^\tau][I - X(X^\tau X)^{-1} X^\tau] = [I - X(X^\tau X)^{-1} X^\tau] = P. \tag{1.25}$$

Lako se dokaže da su svojstvene vrijednosti od P 1 ili 0: Za svojstvenu vrijednost λ postoji pridruženi svojstveni vektor $x \neq 0$ takav da je $Ax = \lambda x$, pa je:

$$A^2 x = AAx = A\lambda x = \lambda Ax.$$

iz čega proizlazi da je $\lambda Ax - Ax = 0$, odnosno:

$$(\lambda - 1)Ax = 0 \implies (Ax = \lambda x) \implies (\lambda - 1)\lambda x = 0 \implies (\lambda - 1)\lambda = 0.$$

Dakle, vrijednost λ može biti samo 0 ili 1.
Matrica $X(X^\tau X)^{-1} X^\tau$ je također idempotentna, pa imamo:

$$\begin{aligned} \operatorname{rank}(X(X^\tau X)^{-1} X^\tau) &= \operatorname{tr}(X(X^\tau X)^{-1} X^\tau) \\ &= \operatorname{tr}(X^\tau X(X^\tau X)^{-1}) = \operatorname{tr}(I_p) = p. \end{aligned} \tag{1.26}$$

S obzirom na to da je $\operatorname{tr}(A - B) = A - B$ imamo:

$$\begin{aligned} \operatorname{rank}(I - X(X^\tau X)^{-1} X^\tau) &= \operatorname{tr}(I - X(X^\tau X)^{-1} X^\tau) \\ &= \operatorname{tr}(I_p) - \operatorname{tr}(X^\tau X(X^\tau X)^{-1}) = n - p. \end{aligned} \tag{1.27}$$

Rezidual sume kvadrata u višestrukoj linearnoj regresiji je $e^\tau e$, a može se zapisati i kao kvadratna forma vektora y:

$$e^\tau e = (y - Xb)^\tau (y - Xb) = y^\tau (I - X(X^\tau X)^{-1} X^\tau) y. \tag{1.28}$$

Koristeći rezultat matematičkog očekivanja kvadratne forme dobivamo:

$$\begin{aligned} \mathrm{E}(e^\tau e) &= \mathrm{E}[y^\tau (I - X(X^\tau X)^{-1} X^\tau) y] \\ &= (X\beta)^\tau (I - X(X^\tau X)^{-1} X^\tau)(X\beta) + \sigma^2 (n - p) \\ &= (X\beta)^\tau (X\beta - X(X^\tau X)^{-1} X^\tau X\beta) + \sigma^2 (n - p) \\ &= \sigma^2 (n - p) \end{aligned}$$

Teorem 1.2.3. Nepristrani procjenitelj varijance u višestrukoj linearnoj regresiji je dan sa:

$$s^2 = \frac{e^\tau e}{n-p} = \frac{y^\tau(I - X(X^\tau X)^{-1}X^\tau)y}{n-p} = \frac{1}{n-p}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2. \tag{1.29}$$

Definicija 1.2.2. Neka je S p-dimenzionalni potprostor vektorskog prostora V. Tada se vektori iz V mogu projicirati na S. Ako potprostor S ima ortonormalnu bazu $(w_1, w_2, \ldots, w_p)$, za svaki vektor $\boldsymbol{y}$ iz V projekcija od y na potprostor S je:

$$P_S\boldsymbol{y} = \sum_{i=1}^{p}(\boldsymbol{y} \cdot w_i)w_i. \tag{1.30}$$

Neka su vektorski prostori S i T dva potprostora vektorskog prostora V i neka vrijedi $S \cup T = V$. Ako za bilo koji vektor $x \in S$ i bilo koji vektor $y \in T$ vrijedi $x \cdot y = 0$, onda se za S i T kaže da su ortogonalni. Možemo reći da je T ortogonalni prostor od S i označiti ga s $T = S^\perp$ te možemo zapisati da je $V = S \cup S^\perp$. Svaki vektor $\boldsymbol{y}$ iz V se može jedinstveno zapisati kao $\boldsymbol{y_S} + \boldsymbol{y}_S^\perp$, za $\boldsymbol{y_S} \in S$ i $\boldsymbol{y}_S^\perp \in S^\perp$.

Projekcija vektora na linearni prostor S je zapravo linearna transformacija vektora i može se prikazati kao projekcijska matrica puta vektor. Projekcijska matrica P je $n \times n$ kvadratna matrica koja daje projekciju sa $\mathbb{R}^n$ na potprostor S.

Teorem 1.2.4. Kvadratna matrica P je projekcijska matrica ako i samo ako je idempotentna, tj. $P^2 = P$.

Teorem 1.2.5. Neka je $U = (u_1, u_2, \ldots, u_k)$ ortonormalna baza za potprostor W od linearnog prostora V. Matrica UU^τ je projekcijska matrica od V na W, tj. za bilo koji vektor $v \in V$ projekcija od v na W je $Proj_W v = UU^\tau v$.

Ako ne krenemo sa ortonormalnom bazom od W, svejedno možemo izračunati projekcijsku matricu, što je sažeto u sljedećem teoremu.

Teorem 1.2.6. Neka je $A = (a_1, a_2, \ldots, a_k)$ bilo koja baza za potprostor W od V. Matrica $A(A^\tau A)^{-1}A^\tau v$ je projekcijska matrica od V na W, tj. za bilo koji vektor $v \in V$ projekcija od v na W je:

$$Proj_W v = A(A^\tau A)^{-1}A^\tau v. \tag{1.31}$$

Lema 1.2.1. Pretpostavimo da je A $n \times k$ matrica čiji su stupci linearno nezavisni. Tada je AA^τ invertibilna.

Lako se zaključi da je projekcijska matrica za W $A(A^\tau A)^{-1}A^\tau$. Projekcijska matrica će biti korisna u sljedećim diskusijama regresijskog modela $Y = X\beta + \varepsilon$.

Konstruira se kvadratna matrica $P = X(XX^\tau)^{-1}X^\tau$. Lako se pokaže da je P idempotentna matrica:

$$P^2 = X(XX^\tau)^{-1}X^\tau X(XX^\tau)^{-1}X^\tau = P.$$

Dakle, P je projekcijska matrica. Matrica $I - P$ je također idempotentna te projekcijska:

$$(I - P)^2 = I - 2P + P^2 = I - 2P + P = I - P.$$

Ove projekcijske matrice se koriste za dobivanje najboljeg linearnog nepristranog procjenitelja (engl. *best linear unbiased estimator*, BLUE).

Definicija 1.2.3. Procjenitelj $T_n = f(X_1, ..., X_n)$ za funkciju gubitka L je najbolji linearni nepristrani procjenitelj ako je linearan, nepristran i ako u klasi svih nepristranih linearnih procjenitelja za L ima najmanju varijancu.

Jedan od najpoznatijih rezultata statistike kaže da procjenitelji β dobiveni metodom najmanjih kvadrata imaju najmanju varijancu među svim linearnim nepristranim procjeniteljima. Taj rezultat se zove Gauss-Markovljev teorem [2, 5].

Teorem 1.2.7 (Gauss-Markov)**.** Neka je $\hat{\theta}$ procjenitelj dobiven metodom najmanjih kvadrata za parametre linearnog regresijskog modela. Ako vrijede Gauss-Markovljevi uvjeti, tada je $\hat{\theta}$ najbolji linearni nepristrani procjenitelj.

Dokaz. Neka je $\hat{\theta} = a^\tau b = a^\tau(X^\tau X)^{-1}X^\tau y$ procjena dobivena metodom najmanjih kvadrata za $a^\tau b$. Neka je $\tilde{\theta} = c^\tau y$ neki drugi linearni nepristrani procjenitelj od $a^\tau b$. Neka je $d^\tau = c^\tau - a^\tau(X^{-1}X)^{-1}X^\tau$. Kako je $c^\tau y$ nepristran, vrijedi

$$\begin{aligned} \mathrm{E}(c^\tau y) &= \mathrm{E}\left(a^\tau(X^T X)^{-1}X^\tau + d^\tau\right) y \\ &= a^\tau\beta + d^\tau X\beta \\ &= a^\tau\beta \end{aligned}$$

iz čega slijedi da je $d^\tau X = 0$. Varijanca procjenitelja je:

$$\begin{aligned}
\mathrm{Var}(c^\tau y) &= c^\tau \mathrm{Var}(y) c \\
&= \sigma^2 c^\tau c \\
&= \sigma^2 \left(a^\tau (X^T X)^{-1} X^\tau + d^\tau\right) \left(a^\tau (X^\tau X)^{-1} X^\tau + d^\tau\right)^\tau \\
&= \sigma^2 \left(a^\tau (X^\tau X)^{-1} X^\tau + d^\tau\right) \left(X (X^T X)^{-1} a + d\right) \\
&= \sigma^2 \left(a^\tau (X^\tau X)^{-1} X^\tau X (X^\tau X)^{-1} a + a^\tau (X^\tau X)^{-1} \underbrace{X^\tau d}_{=0} + \underbrace{d^\tau X}_{=0} (X^\tau X)^{-1} a + d^\tau d\right) \\
&= \sigma^2 \left(\underbrace{a^\tau (X^\tau X)^{-1} a}_{\mathrm{Var}(\hat{\theta})} + \underbrace{d^\tau d}_{\geq 0}\right)
\end{aligned}$$

Iz toga slijedi da je $\mathrm{Var}(\hat{\theta}) \leq \mathrm{Var}(\tilde{\theta})$ za sve ostale linearne nepristrane procjenitelje $\tilde{\theta}$. □

1.3 Gradijentni spust

Gradijentni spust je iterativni optimizacijski algoritam za pronalaženje minimuma funkcije. Algoritam je popularan u području strojnog učenja gdje se često koristi za pronalaženje minimalne greške tako da minimizira funkciju gubitka (engl. *cost function*) koja se još zove i funkcija kvadratne pogreške.

Definiramo funkciju gubitka kao:

$$J(\beta_0, \ldots, \beta_{k-1}) = J(\boldsymbol{\beta}) = \frac{1}{2n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \tag{1.32}$$

Algoritam gradijentnog spusta za $n \geqslant 1$:

Algoritam 1.1 Gradijentni spust.

1: **dok** nema konvergencije **ponavljaj**
2: **za** $n \leftarrow 1$ **do** k **radi**
3: $\beta_j := \beta_j - \alpha \underbrace{\frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i) x_{ij}}_{\frac{\partial}{\partial \beta_0} J(\beta)}$
4: **kraj petlje**
5: **kraj petlje**

Parametar α je stopa učenja (engl. *learning rate*) i predstavlja veličinu koraka gradijentnog spusta. Ako je α premalen, gradijentni spust može biti spor, a ako je prevelik može doći do toga da se minimum promaši (engl. *overshooting*) te do nemogućnosti konvergencije ili čak do divergencije. Kako se približavamo minimumu, gradijentni spust će automatski uzimati sve manje i manje korake. Smjer u kojem ide gradijentni spust je određen s parcijalnom derivacijom od J. Intuicija iza konvergencije je da kad se parcijalne derivacije od J približavaju nuli, dolazimo do dna naše konveksne funkcije.

Gradijentni spust može konvergirati u lokalni minimum, no ako je funkcija konveksna, kao što je pokazano da jest u slučaju linearne regresije, doći ćemo do globalnog minimuma. Minimum je kod linearne regresije jedinstven i gradijentni spust uvijek konvergira (pretpostavimo li da α nije prevelik). J je konveksna kvadratna funkcija.

Algoritam se može ubrzati i tako da su ulazni podaci otprilike u istom rasponu, pa možemo primijeniti skaliranje varijabli i normalizaciju ako je potrebno. Kad se pitamo radi li gradijentni spust točno, dobro je nacrtati graf funkcije gubitka J s brojem iteracija na x-osi i ako primijetimo da negdje J raste, vjerojatno treba smanjiti α. Također, možemo pronaći neku vrlo malu vrijednost E i deklarirati konvergenciju ako se J smanji za manje od E u jednoj iteraciji. Dokazano je da ako je α dovoljno malen, J će se smanjivati sa svakom iteracijom.

Možemo napraviti usporedbu gradijentnog spusta s rješavanjem normalne jednadžbe 1.18:

Gradijentni spust	**Normalna jednadžba**
treba se izabrati α	ne treba se izabrati α
treba (puno) iterirati	nema iteracija
$O(kn^2)$	$O(n^3)$, treba izračunati inverz od $X^\tau X$
radi dobro za veliki n	radi sporo za veliki n

Postoje različite vrste gradijentnog spusta od kojih su najpoznatiji *batch* i stohastički gradijentni spust. *Batch* spust računa gradijent koristeći cijeli skup podataka i dobro radi za konveksne ili relativno glatke mnogostrukosti grešaka, za razliku od stohastičkog koji koristi jedan uzorak (ili češće *minibatch* od nekoliko uzoraka) koji radi bolje za mnogostrukosti grešaka koji imaju mnogo lokalnih minimuma/maksimuma. Stohastički pristup ima više šuma, pa se zato češće koristi *minibatch* koji reducira određeni dio šuma i svejedno uspije izbjeći lokalne minimume. Također je i puno brži za računanje od *batch*-a.

Poglavlje 2

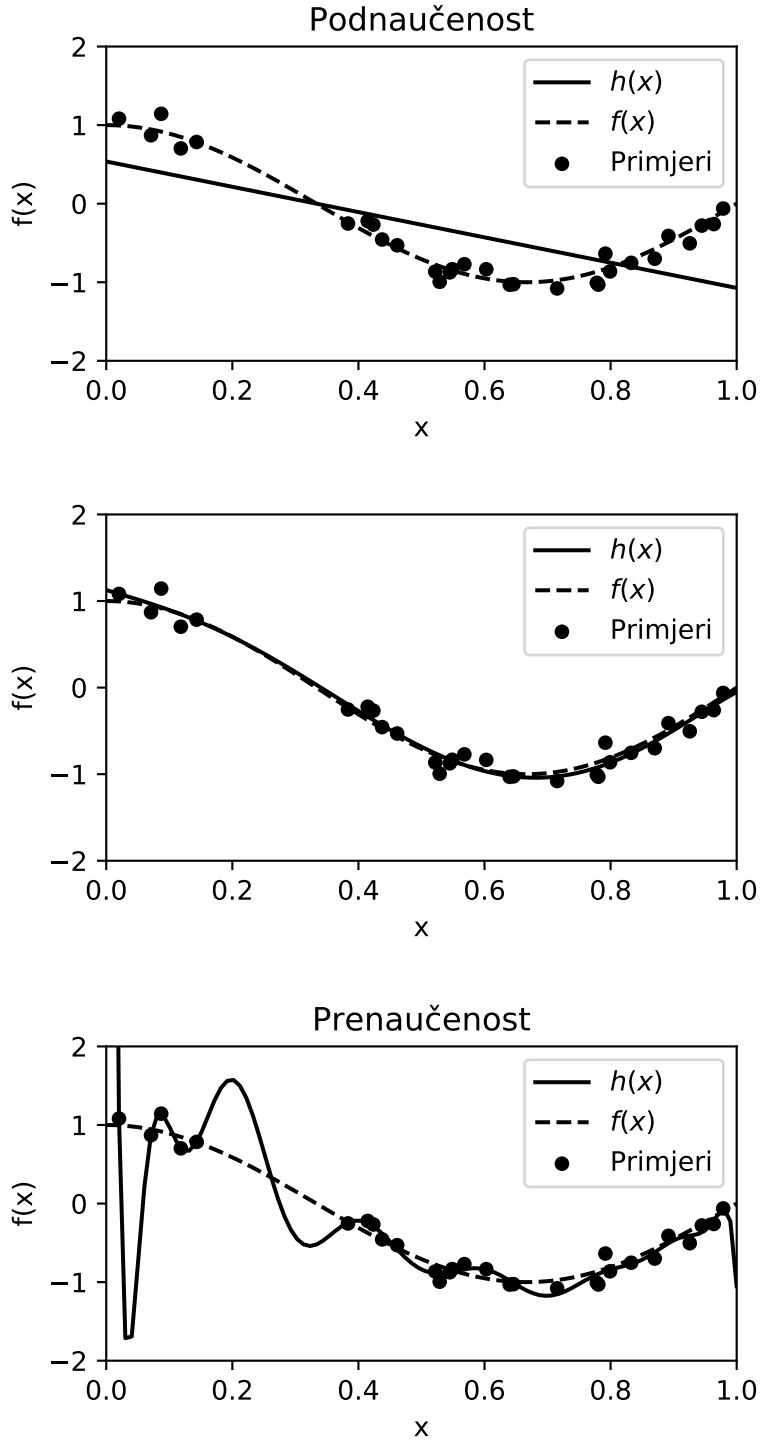
Stabla odluke

Kod problema nadziranog učenja (engl. *supervised learning*) imamo skup podataka i već znamo kako bi trebao izgledati točan izlaz za razliku od nenadziranog učenja. Problemi nadziranog učenja su kategorizirani u regresijske i klasifikacijske probleme. U ovom radu promatrat će se regresijski problem u kojem pokušavamo predvidjeti izlaz koji je neprekidna varijabla, za razliku od klasifikacijskog koji ima diskretan izlaz.

Postoje različiti klasifikatori poput neuronskih mreža, stroja potpornih vektora (engl. *support vector machine*, SVM), najbližih susjeda (engl. *nearest neighbor*) itd. Freund i Schapire [13] su pokazali da se može konstruirati jaki klasifikator od više slabih klasifikatora koji će točno klasificirati sve uzorke. Taj klasifikator bi se trebao sastojati od više slabih kojima su pridodane težine, tj. moć glasanja. Svoju metodu su nazvali *Adaboost* od engl. *adaptive boosting*.

Dio koji slijedi je baziran na predavanjima u [9]. Kod predviđanja izlazne varijable y može doći do problema prenaučenosti (engl. *overfitting*) i podnaučenosti (engl. *underfitting*). Problem je ilustriran na slici 2.1. Podnaučenost ili problem visoke pristranosti (engl. *high bias*) je slučaj kad funkcija loše opisuje podatke i uglavnom je uzrokovana prejednostavnom funkcijom ili funkcijom koja koristi premalo značajki. S druge strane, prenaučenost ili problem visoke varijance (engl. *high variance*) je uzrokovana funkcijom koja odlično opisuje podatke, no radi toga ne predviđa dobro nove podatke. Obično je uzrokovana kompliciranom funkcijom koja stvara mnogo nepotrebnih krivulja nepovezanih s podacima. Cilj je pronaći dobar kompromis između visoke varijance i visoke pristranosti (engl. *bias-variance tradeoff*).

Kako bismo pristupili problemu prenaučenosti, možemo ili smanjiti broj značajki (ručno ili koristeći algoritam za odabir) ili koristiti regularizaciju (radi dobro kad imamo mnogo djelomično korisnih značajki). Kod prenaučenosti, funkcija gubitka na skupu za učenje J_{train} će biti niska, a na unakrsno validiranom skupu J_{CV} će biti



Slika 2.1: Funkcija koja generira podatke je $f(x)$ dok je funkcija koju model otkriva kroz primjere $h(x)$. Što je veći stupanj polinoma to je bolja greška u odnosu na primjere, ali model ne može dobro generalizirati na neviđene primjere (koje generira $f(x)$).

puno veći od J_{train}. Kod podnaučenosti, J_{train} i J_{CV} će biti visoki i $J_{CV} \approx J_{train}$.

Dodavanje regularizacije podrazumijeva dodavanje dodatnog člana $\lambda \sum_{i=1}^{n} {\beta_j}^2$ u funkciju gubitka $J(\beta)$. Dodamo li regularizaciju u linearnu regresiju, regularizacijski parametar λ također utječe na pristranost i varijancu. Ako je λ velik, dolazi do podnaučenosti i regresijski parametri su približno jednaki nuli. Ako je λ blizu nule, dolazi do prenaučenosti. Kako bi odabrali dobar λ trebamo isprobati različite λ i naučiti regresijske parametre i izračunati grešku unakrsne validacije koristeći naučene parametre bez regularizacije i odabrati onaj λ za koji je greška najmanja na skupu unakrsne validacije.

2.1 Regresijska stabla

Regresijska stabla su varijanta stabala odluke koja se mogu koristiti za procjenjivanje realnih funkcija, pa tako i za procjenjivanje parametara linearne regresije. Linearna regresija je globalni model gdje postoji jedna formula za predviđanje za sve podatke. No, kad podaci imaju mnogo značajki koje međusobno djeluju na komplicirane i nelinearne načine, primjenjivanje jednog modela može biti teško.

Alternativni pristup je stoga particionirati prostor na manje dijelove gdje su interakcije lakše izvedive. Metodologija konstrukcije stabla dopušta da ulazne varijable budu mješavina realnih i kategoričkih varijabli. Proces konstrukcije stabla se zove binarno rekurzivno particioniranje (engl. *binary recursive partitioning*), što je iterativni proces koji dijeli podatke u particije ili grane i onda particionira particije u manje grupe. Cilj je doći do manjih dijelova podataka koji su dovoljno dobri da na njih primijenimo jednostavne modele.

Svaki terminalni čvor ili list predstavlja ćeliju particije kojoj je pridružen jednostavni model koji se primjenjuje samo na tu ćeliju. Da bismo znali u kojoj smo ćeliji, počnemo od korijena i pitamo niz pitanja o značajkama. Unutrašnji čvorovi su označeni pitanjima, a grane odgovorima. Model u svakoj ćeliji je konstantna procjena izlazne varijable, tj. pretpostavimo li da su $(x_1, y_1), (x_2, y_2), \ldots, (x_c, y_c)$ svi uzorci koji pripadaju listu l. Onda je naš model za l samo $\hat{y} = \frac{1}{c} \sum_{i=1}^{c} y_i$, uzoračka srednja vrijednost izlazne varijable u toj ćeliji. Takav model brzo predviđa, gledanjem u stablo se lako vidi koje su varijable bitne za predviđanje, ako neki podaci nedostaju svejedno se može napraviti predviđanje i postoje brzi i pouzdani algoritmi za učenje tih stabala.

Na početku su svi podaci iz skupa za učenje u istoj particiji. Algoritam zatim počinje alocirati podatke u prve dvije particije ili grane, koristeći svaki mogući binarni rez na svakom području. Algoritam uzima rez koji minimizira sumu kvadrata devijacija od srednje vrijednosti u dvije odvojene particije. Na isti način se režu i nove grane. Proces se nastavlja dok svaki čvor ne postane dovoljno malen i postane list (ili suma kvadrata devijacija postane nula).

S obzirom na to da se stablo konstruira iz skupa za učenje, potpuno razvijeno stablo obično ima problem prenaučenosti, što uzrokuje loše predviđanje stvarnih podataka. Zato se stablo podrezuje (engl. *pruning*) koristeći validacijski skup. Time se minimizira suma varijance izlazne varijable u validacijskim podacima (uzevši jedan po jedan list) i minimizira se suma produkta faktora složenosti troškova (engl. *cost complexity factor*) i broja listova. Ako je faktor složenosti troškova nula, onda podrezivanje znači jednostavno pronalaženje stabla koje je najučinkovitije na validacijskim podacima u smislu ukupne varijance listova. Veće vrijednosti faktora složenosti troškova će rezultirati manjim stablima. Podrezivanje se primjenjuje tako da čvor koji je zadnji izrastao je prvi na redu za eliminaciju.

Algoritam regresijskih stabala se može koristiti za pronalaženje jednog modela koji dobro predviđa nove podatke, no ne možemo znati postoji li bolji prediktor. Međutim, postoji nekoliko ansambl (engl. *ensemble*) metoda koje se koriste s regresijskim stablima: *bagging*, *boosting* i slučajna stabla. Ansambl nastane kombiniranjem više slabijih modela koji kombiniranjem stvore novi, točni, jaki model. Slabim modelom smatra se klasifikator koji je samo nešto bolji od slučajnog pogađanja, tj. slabo je malo koreliran sa stvarnom klasifikacijom. Stvaranjem više različitih regresijskih modela i uzimajući različite uzorke originalnog skupa podataka i kombiniranjem njihovih izlaza efikasno se smanjuje varijanca i poboljšava model. *Bagging*, *boosting* i slučajna stabla se razlikuju u selekciji skupa za učenje, u generiranju slabih modela i u kombiniranju izlaza. No, u sve tri metode svaki slabi model se uči na cijelom skupu za učenje tako da postane izvrstan u nekom dijelu skupa podataka.

Bagging ili *bootstrap* agregiranje je bio jedan od prvih ansambl algoritama. *Bagging* generira nekoliko skupova za učenje koristeći slučajno uzorkovanje sa zamjenama, primjenjuje algoritam regresijskih stabala na svaki podatkovni skup i onda uzima prosjek između modela za izračun predviđanja za nove podatke. *Bagging* je moguće paralelizirati relativno lako, što ga čini boljim izborom za vrlo velike podatkovne skupove.

Boosting izgrađuje snažan model tako da se tijekom učenja u skup modela dodaju oni modeli koji su učeni na primjerima na kojima trenutni skup modela dobiva netočna predviđanja.

2.2 Gradijentni *boosting*

Algoritam gradijentnog *boosting*-a je jedan od najmoćnijih tehnika za izgradnju modela za predviđanje. U statističkom smislu, *boosting* možemo shvatiti kao numerički problem optimizacije gdje je cilj minimizirati gubitak modela dodavanjem slabih klasifikatora koristeći algoritam sličan gradijentnom spustu. Ova vrsta algoritma spada

u stadijske (engl. *stagewise*) aditivne modele, što znači da se dodaje jedan po jedan slabi klasifikator, a ostali ostaju nepromijenjeni.

Na svakom stadiju s, $1 \leq s \leq S$ imamo nesavršeni model m_s. Na početnom stadiju to može biti model koji uvijek daje srednju vrijednost c (konstanta) ciljne varijable y. Algoritam gradijentnog *boosting*-a izgrađuje poboljšanje modela m_s dodavanjem procjenitelja h – $m_{s+1}(x) = m_s(x) + h(x)$. Za pronalazak tog procjenitelja koristi se pretpostavka da on može biti savršen:

$$m_{s+1}(x) = m_s(x) + h(x) = y \ ,$$

iz čega slijedi $h(x) = y - m_s(x)$. Postupak gradijentnog *boosting*-a u tom slučaju uči procjenitelja h na rezidualu $y - m_s(x)$ koji je jednak negativnom gradijentnu kvadratne pogreške $\frac{1}{2}(y - m_s(x))^2$ – zbog toga je u nazivu *gradijentni*, svaki novi model u sumi bi trebao minimizirati pogrešku $L(y, m_s(x) + h(x))$. Algoritam je baziran na sljedećim jednadžbama:

$$m_0(x) = \underset{c}{\operatorname{argmin}} \sum_{i=1}^{n} L(y_i, c),$$

$$m_s(x) = m_{s-1}(x) + \underset{h \in \mathcal{H}}{\operatorname{argmin}} \sum_{i=1}^{n} L(y_i, m_{s-1}(x_i) + h(x_i)).$$

Ako koristimo stabla kao dio aditivnog modela $m_s(x)$, onda se za pronalazak procjenitelja h koristi neki od algoritama za učenje stabala. Generalniji algoritam gradijentnog *boosting*-a vidi se u pseudokodu 2.1.

Algoritam 2.1 Algoritam gradijentnog *boosting*-a

Potrebno: Skup podataka $\{(x_i, y_i)\}_{i=1}^n$ i broj prolaza S.

$m_0(x) \leftarrow \underset{c}{\operatorname{argmin}} \sum_{i=1}^{n} L(y_i, c)$
za svaki $s \in \{1, 2, \ldots, S\}$ **radi**
    za svaki $i \in \{1, 2, \ldots, n\}$ **radi**
        Izračunaj rezidual $r_{si} = -\left[\frac{\partial L(y_i, m(x_i))}{\partial m(x_i)}\right]_{m(x)=m_{s-1}(x)}$
    kraj petlje
    Nauči $h(x)$ koristeći skup $\{(x_i, r_{si})\}_{i=1}^n$
    $c \leftarrow \underset{c}{\operatorname{argmin}} \sum_{i=1}^{n} L(y_i, m_{s-1}(x) + c \cdot h(x))$
    Ažuriraj model $m_s(x) \leftarrow m_{s-1}(x) + c \cdot h(x)$
kraj petlje
vrati $m_S(x)$

Ako koristimo stabla za procjenitelj h, onda je moguće tu informaciju iskoristiti i tražiti konstantni c za svaku regiju koju određuje list stabla. Friedman naziva taj

algoritam *TreeBoost* [1]. Pronalazak te konstante ne treba biti srednja vrijednost y_i na tom intervalu, nego može biti i medijan. Time bi se minimizirala funkcija $|y - h(x)|$. Algoritam je prikazan na pseudokodu 2.2.

Algoritam 2.2 Algoritam gradijentnog *boosting*-a sa stablima

Potrebno: Skup podataka $\{(x_i, y_i)\}_{i=1}^n$ i broj prolaza S.
1: $m_0(x) \leftarrow \underset{c}{\operatorname{argmin}} \sum_{i=1}^n L(y_i, c)$
2: **za svaki** $s \in \{1, 2, \ldots, S\}$ **radi**
3: **za svaki** $i \in \{1, 2, \ldots, n\}$ **radi**
4: Izračunaj rezidual $r_{si} = -\left[\frac{\partial L(y_i, m(x_i))}{\partial m(x_i)}\right]_{m(x)=m_{s-1}(x)}$
5: **kraj petlje**
6: Nauči stablo koristeći skup $\{(x_i, r_{si})\}_{i=1}^n$ za regije R_{js} završnih čvorova iz J.
7: **za svaki** $s \in \{1, 2, \ldots, |J|\}$ **radi**
8: $c_{js} \leftarrow \underset{c}{\operatorname{argmin}} \sum_{x_i \in R_{js}}^n L(y_i, m_{s-1}(x) + c)$
9: **kraj petlje**
10: Ažuriraj model $m_s(x) \leftarrow m_{s-1}(x) + \sum_{j=1}^{|J|} c_{js} \cdot I(x \in R_{js})$
11: **kraj petlje**
12: **vrati** $m_S(x)$

2.3 Algoritam DART

Rashmi i Gilad-Bachrach opisuju algoritam DART [15]. Nadograđujući se na ideje Friedmanovih algoritama opisuju poboljšanje koje zaobilazi uobičajene mane prijašnjih pristupa. Tijekom stadijskog učenja pojavljuje se problem u kojem stabla dodana na kasnijim stadijima utječu na predviđanje samo nekoliko primjera iz skupa za učenje, a za ostale primjere nisu dovoljno učinkovita. Prije se koristio postupak *shrinkage* za rješavanje tog problema, ali on produžuje trajanje algoritma učenja i predviđanja. Umjesto ažuriranja modela u liniji 8 pseudokoda 2.1 koristi se pravilo:

$$m_s(x) \leftarrow m_{s-1}(x) + \nu \cdot c \cdot h(x),$$

gdje je ν zvan parametrom brzine učenja (engl. *learning rate*). Rashmi i Gilad-Bachrach u svojim rezultatima pokazuju da problem i dalje ostaje, samo je efekt manji.

U poglavlju 4 koristi se model naučen algoritmom DART. Algoritam je prikazan u pseudokodu 2.3. Ideja je prvi put korištena kod učenja neuronskih mreža, gdje bi se pojedinačni skriveni slojevi ili čvorovi isključili iz ažuriranja u trenutnoj iteraciji.

Algoritam 2.3 Algoritam DART

N je broj stabala u ansamblu.
$S_1 \leftarrow \{x, -L'_x(0)\}$
Neka je T_1 stablo učeno na skupu S_1
$M \leftarrow \{T_1\}$
za svaki $t \in \{2, \ldots, N\}$ **radi**
    $D \leftarrow$ podskup od M takav da $T \in M$ je član od D s vjerojatnošću p_{drop}
    ako $D = \emptyset$ **onda**
        $D \leftarrow$ slučajan član iz M
    kraj
    $\hat{M} \leftarrow M \setminus D$
    $S_t \leftarrow \left\{x, -L'_x\left(\hat{M}(x)\right)\right\}$
    Neka je T_t stablo učeno na skupu S_t
    $M \leftarrow M \cup \left\{\frac{T_t}{|D|+1}\right\}$
    $M \leftarrow \left\{\frac{|D|}{|D|+1}T \mid T \in M \cap D\right\}$
kraj petlje
vrati M

U ovom algoritmu odabire se podskup stabala D takav da je element $T \in M$ izabran vjerojatnošću p_{drop}. Onda se novo stablo uči bez elemenata u skupu D i u M se ta stabla dodaju s novim težinama.

Poglavlje 3

Opisna statistika

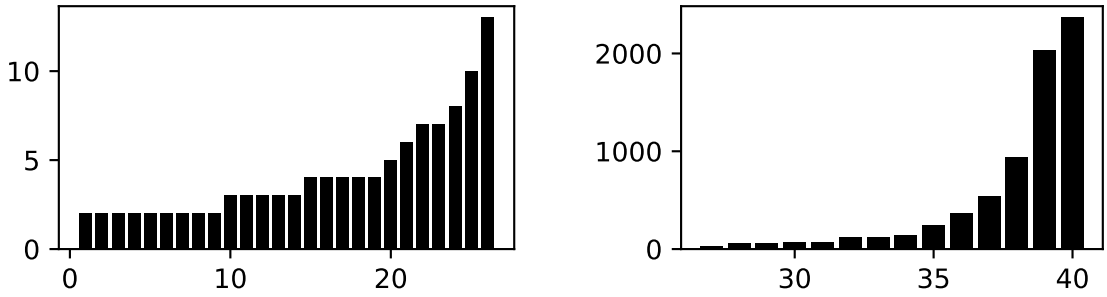
U ovom poglavlju je prisutan opis podataka pomoću kojih se radio model te njihova opisna statistika. Istaknute su varijable koje su bitne u modeliranju. Izvorni kod za konstrukciju i stvaranje slika prisutan je u dodatku C.

3.1 Opis podataka

U podacima je sadržano $N = 13102$ redaka i $k = 82$ stupca. Svaki redak sadrži informacije o marketinškoj promociji jednog proizvoda. Svaki stupac predstavlja podatak (varijablu) o promociji. Stupci sadrže kategoričke i kontinuirane vrijednosti. Kategorički stupci sadrže identifikator proizvoda, mjera količine proizvoda u pakiranju (komad, paket, gram, mililitar i dr.), vrsta prodavaonice u kojoj se proizvod prodaje (ljekarna, supermarket, diskont i dr.), vrsta promocije (postotak popusta, kupon, dodatno besplatno pakiranje, popust na količinu i dr.), medij kojim se oglašava (društvene mreže, dnevne novine, letci, video prilozi i dr.), tip proizvoda (juhe, slatkiši i dr.), rasprostranjenost promocije (lokalno, regija, više regija ili cijela zemlja), brend proizvoda, zemlja u kojoj je aktivna promocija, trgovački lanac itd. Kontinuirani stupci sadrže cijenu prije i za vrijeme promocije, trajanje promocije u danima, stranica u promotivnom materijalu, ukupan broj stranica u promotivnom materijalu i dr.

Od 82 stupca 52 je kategoričkih, a 30 kontinuiranih. Neke koji su uzeti kao kategorički možemo interpretirati i kao kontinuirane (godina promocije, mjesec promocije i sl.). Stupac koji predstavlja ciljnu varijablu y je kontinuiran. Ciljna varijabla y predstavlja ishod promocije – prihodi u određenoj valuti. U poglavlju 4 modeliran je odnos ishoda y i značajki promocije. Ako promocija ima velik utjecaj, onda bi model mogao biti dosta precizan.

Neki od stupaca su suvišni za modeliranje. Stupci poput identifikatora proizvoda,



Slika 3.1: Prikazan je broj jedinstvenih vrijednosti za svaku kategoričku varijablu (rastući poredak). Ukupno je 26 kategoričkih varijabli ispod 20 jedinstvenih vrijednosti (lijeva) i 14 iznad (desna). Na apscisi je broj varijable u danom poretku, a na ordinati je broj jedinstvenih vrijednosti.

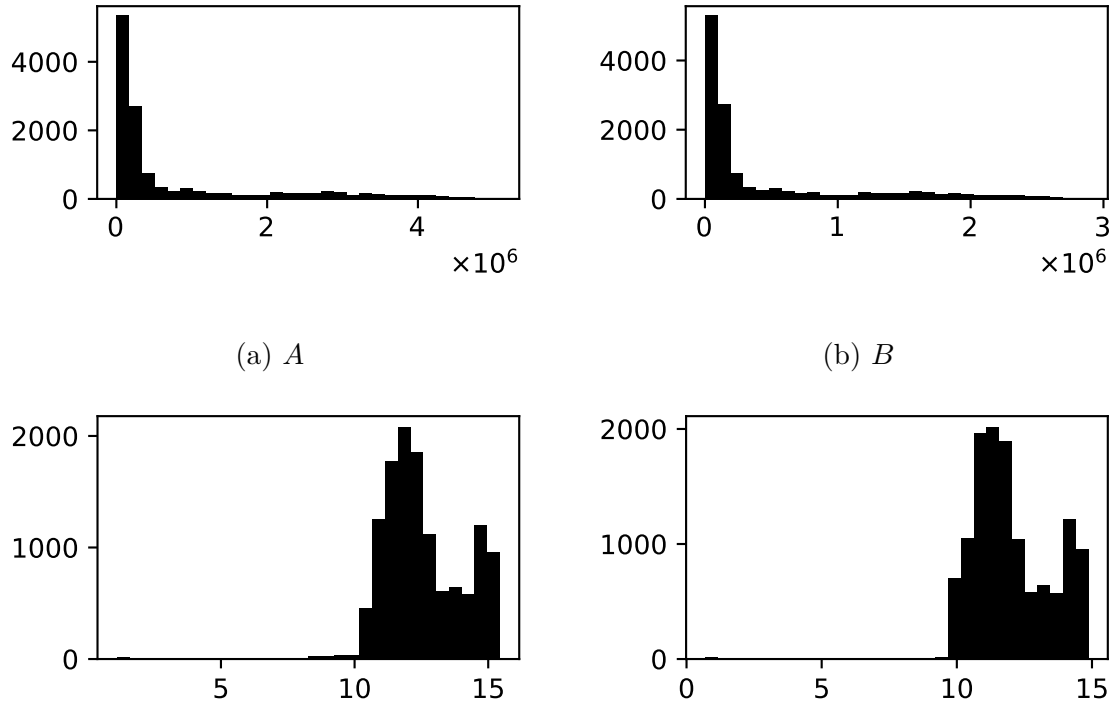
web poveznica do artikla i datum od kad do kad se vodila promocija su previše detaljni i ne daju nikakve informacije (broj jedinstvenih vrijednosti je jako velik). Neki su prazni ili sadrže samo jednu vrijednost (zemlja promocije je samo Hrvatska), a neki su ekvivalentni drugim stupcima (prikazuju iste podatke samo na drugačiji način – npr. tekst medija kojim se oglašava i brojčani identifikator medija, obje varijable su kategoričke i histogrami su im identični).

Nakon eliminacije ostaje $k = 57$ stupaca od kojih je kategoričkih 40, kontinuiranih 16 i jedna ciljna varijabla. U podacima postoje redci gdje je ciljna varijabla prazna stoga je i njih potrebno odbaciti. Nakon toga ostaje $N = 12627$ redaka. Za daljnju analizu koriste se upravo količina redaka i stupaca navedena u ovom odlomku (osim ako nije navedeno drugačije).

3.2 Analiza pojedinačnih varijabli

Na slici 3.1 prikazan je broj jedinstvenih vrijednosti za svaku kategoričku varijablu. Neke kategoričke varijable imaju previše jedinstvenih vrijednosti. Ako bismo koristili kodiranje opisano u potpoglavlju 4.1, onda bi vektor značajki koji kodira samo varijablu s najviše jedinstvenih vrijednosti imao dimenziju za nekoliko tisuća veću. Takvo povećanje dimenzionalnosti zahtijevalo bi sofisticiranije modele.

Slika 3.2 prikazuje dvije varijable iz podataka koje, kao i ciljna varijabla, pokazuju eksponencijalnu distribuciju. U potpoglavlju 3.3 se dodatno analizira njihov odnos s ciljnom varijablom (zaključak je da postoji jaka korelacija te je pretpostavka da su varijable dobri prediktori ciljne). Postoje varijable poput cijene proizvoda, težine



(a) A (b) B

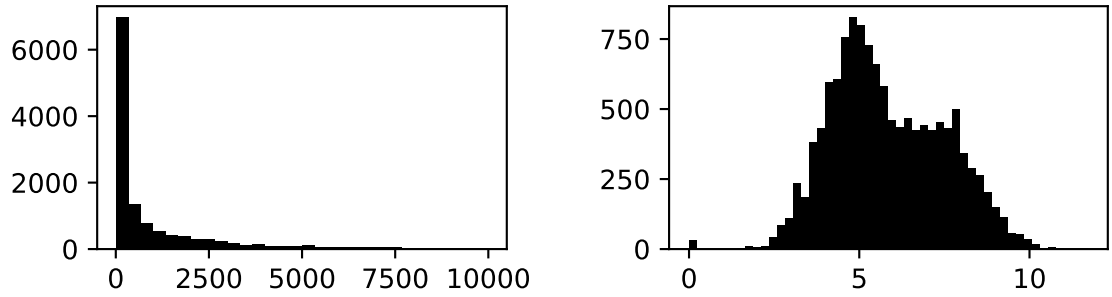
(c) $\log A$ (d) $\log B$

Slika 3.2: Zakon potencije dvije varijable iz podataka.

ili količine za koje se može reći da također imaju eksponencijalnu distribuciju (puno jeftinih, laganih proizvoda, a sve manje skupljih i teških) koje su uključene u analizu u sljedećem potpoglavlju.

3.3 Utjecaj na ciljnu varijablu

Histogram ciljne varijable y prikazan je na slici 3.3. Na linearnoj skali prikazanoj na slici 3.3a može se uočiti zakon potencije (engl. *power law*). Kako vrijednost ciljne varijable raste tako sve manje promocija dostiže tu vrijednost. Zbog jasnijeg prikaza analize u nastavku ovog potpoglavlja koristi se logaritmirana vrijednost ciljne varijable prikazana na slici 3.3b (raspon vrijednosti je manji). Svi prikazani dijagrami pravokutnika u ovom potpoglavlju prikazuju podatke gdje je vrijednost ciljne varijable logaritmirana, a korišteni statistički testovi su također primijenjeni na logaritmirane vrijednosti ciljne varijable. Deklaracija statističke značajnosti dana je na razini zna-



(a) (b)

Slika 3.3: Prikazana je ciljna varijabla y (histogram) u linearnoj i logaritamskoj skali. Za prikaz na linearnoj korišteni su podaci manji od 10000 (inače bi maksimum bio preko sto tisuća) i 30 razreda, a za drugu 50 razreda.

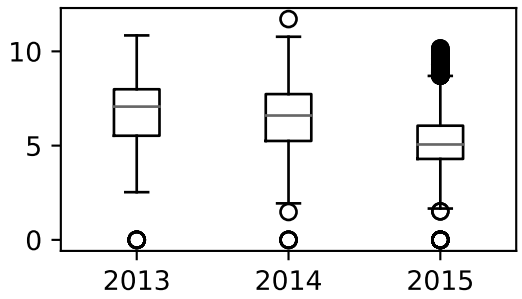
čajnosti $\alpha = 0.05$ (hipoteza jednakosti je odbačena ako je $p < 0.05$). Pretpostavka o distribuciji ciljne varijable s obzirom na grafički prikaz mogla je uključivati eksponencijalnu ili lognormalnu distribuciju. Statistički značajna jednakost za te distribucije nije postignuta. Moguće je pretpostaviti iz slike 3.3b da je slučajna varijabla zbroj dvije lognormalne distribucije i koristiti prikladne testove, ali radi jednostavnosti u nastavku se koriste neparametarski statistički testovi.

Slike 3.4 i 3.5 prikazuju ovisnost ciljne varijable o pojedinačnim kategorijama varijabli. Kod slika 3.4a i 3.5b sve kategorije se statistički značajno ($p < 0.05$) razlikuju jedna od drugih (distribucije im nisu iste) – korišten je Kolmogorov-Smirnov test. Za vrednovanje bi bilo zanimljivo izbaciti podatke od buduće godine i probati naći model koji dobro predviđa tu godinu uzevši podatke od prošlih. S obzirom na to da ciljna varijabla nije isto distribuirana podatkovni skupovi koji se koriste za treniranje moraju uključivati podatke sve tri godine. Na slici 3.4c može se vidjeti sličnost između promocija koje traju 3, 4 i 21 dan. Sve su statistički značajno identične.

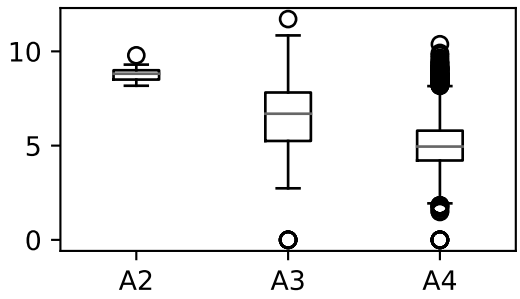
Veličina promotivnog materijala bi trebala imati utjecaj na uspješnost promocije. Na slici 3.4b to se može i vidjeti: što je veći papir to je veća i ciljna varijabla. Korištenjem jednostranog Mann–Whitney U testa može se zaključiti da je format A2 uspješniji od formata A3 i A4, te je A3 uspješniji od A4.

Slika 3.8 prikazuje deset proizvoda koji imaju najviše promocija (raspon je od 200 do 400 promocija po proizvodu). Unatoč tome što su promocije za isti proizvod, vrijednost ciljne varijable varira i statistički je značajno različita za skoro svaki par proizvoda. Ovaj rezultat upućuje na to da bi identifikator proizvoda bilo korisno uključiti u reprezentaciju vektora stupca.

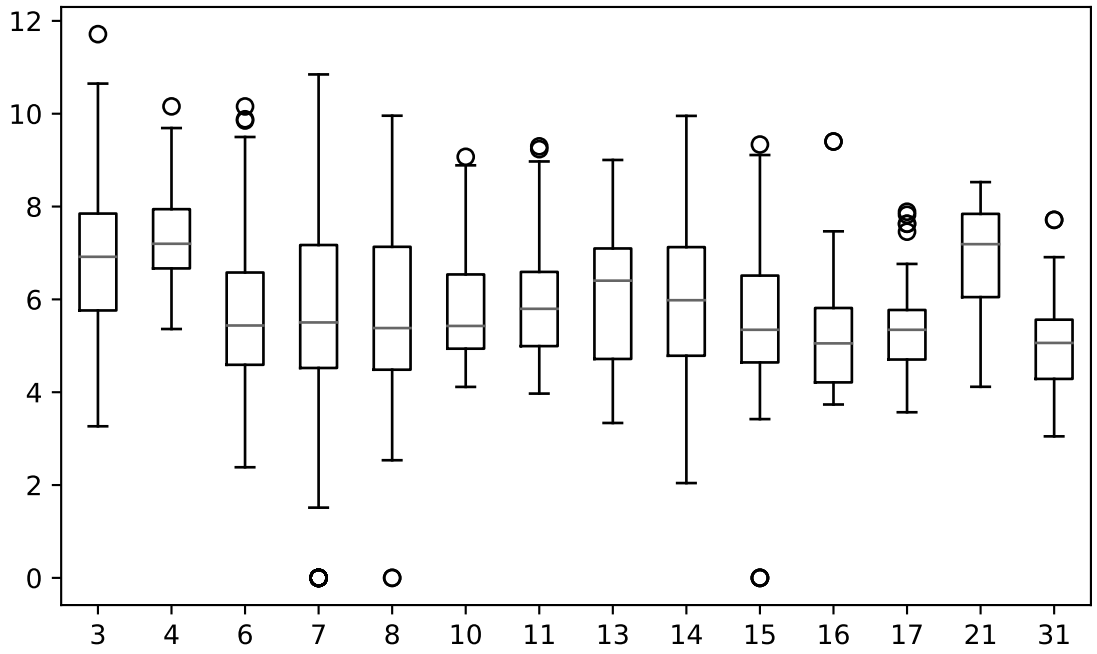
Zanimljivo je pogledati ovisnost ciljne varijable o broju stranice na kojem se oglas



(a) Godina

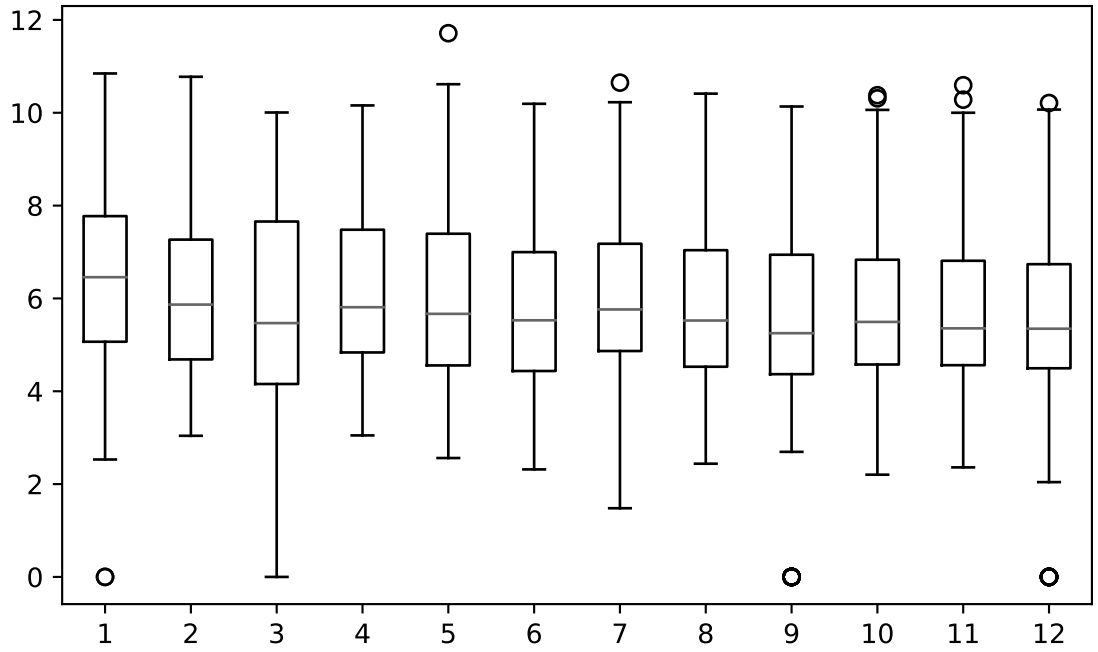


(b) Format stranice

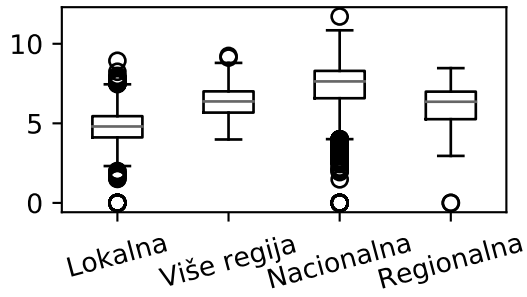


(c) Trajanje promocije u danima

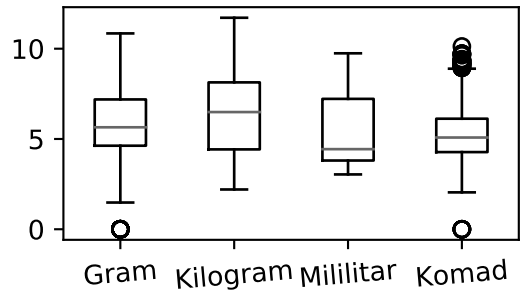
Slika 3.4: Slike prikazuju kategoričke varijable i ponašanje ciljne varijable za svaku pojedinu kategoriju. Tip kategoričke varijable piše ispod pojedinačne slike. Vrijednost ciljne varijable na y-osi je na logaritamskoj skali (baza e). Slike su u obliku dijagrama pravokutnika (engl. *box plot*).



(a) Mjesec u godini

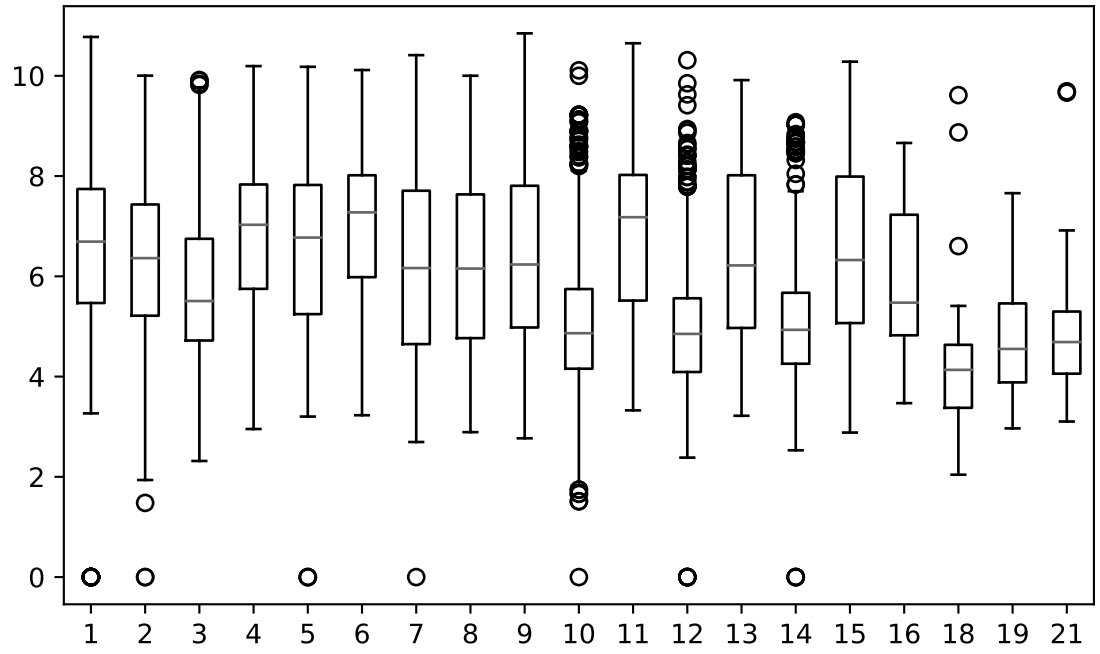


(b) Rasprostranjenost promocije



(c) Mjerna jedinica proizvoda

Slika 3.5: Slike prikazuju kategoričke varijable i ponašanje ciljne varijable za svaku pojedinu kategoriju. Tip kategoričke varijable piše ispod pojedinačne slike. Vrijednost ciljne varijable na y-osi je na logaritamskoj skali (baza e). Slike su u obliku dijagrama pravokutnika (engl. *box plot*).



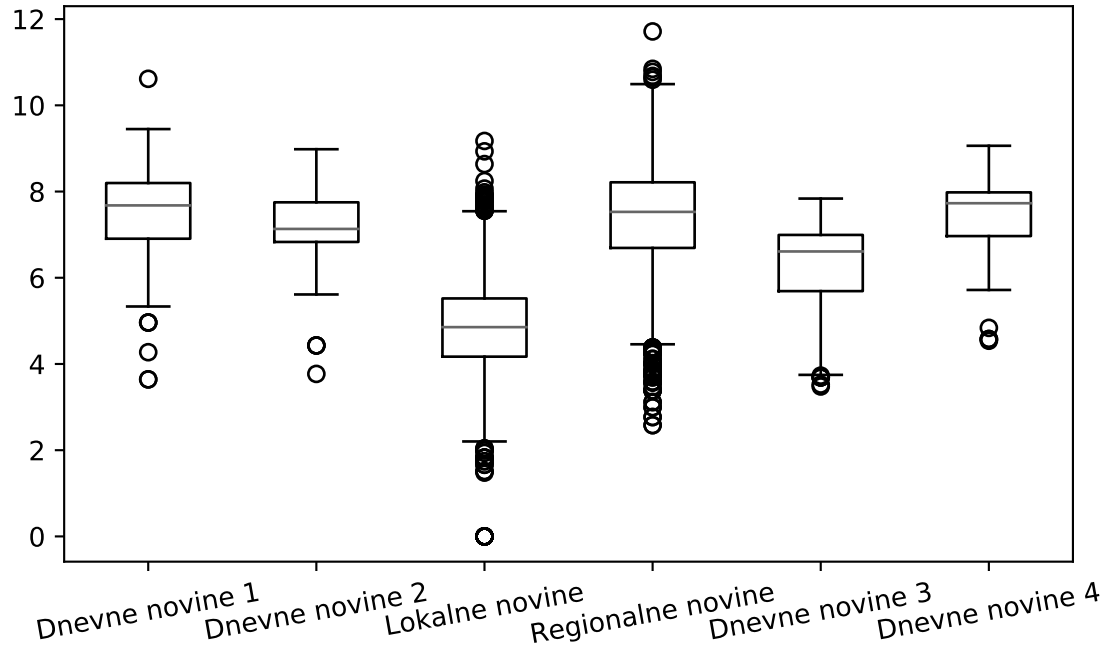
Slika 3.6: Prikazano je ponašanje ciljne varijable u ovisnosti o broju stranice na kojoj se proizvod nalazio na promotivnom materijalu.

za proizvod nalazio – prikazana na slici 3.6. Pretpostavka je da će proizvodi koje se promovira na početnim stranicama imati veću vrijednost ciljne varijable, tj. takve promocije su uspješnije. Korišten je Mann–Whitney U test za usporedbu jesu li vrijednosti ciljne varijable za pojedinačnu stranicu (ili skup stranica) veće od druge. Promocije koje su prisutne na četvrtoj, petoj i šestoj stranici su statistički značajno uspješnije od onih na prvoj, drugoj i trećoj stranici. Promocija koja je bila u prvih deset stranica je statistički značajno uspješnija od promocije koja je na sljedećim stranicama.

Slika 3.7 pokazuje ovisnost ciljne varijable o mediju u kojem je proizvod oglašavan. Statistički značajna razlika postoji između kategorije lokalnog i regionalnog medija te pojedinačnih novinskih dnevnih listova.

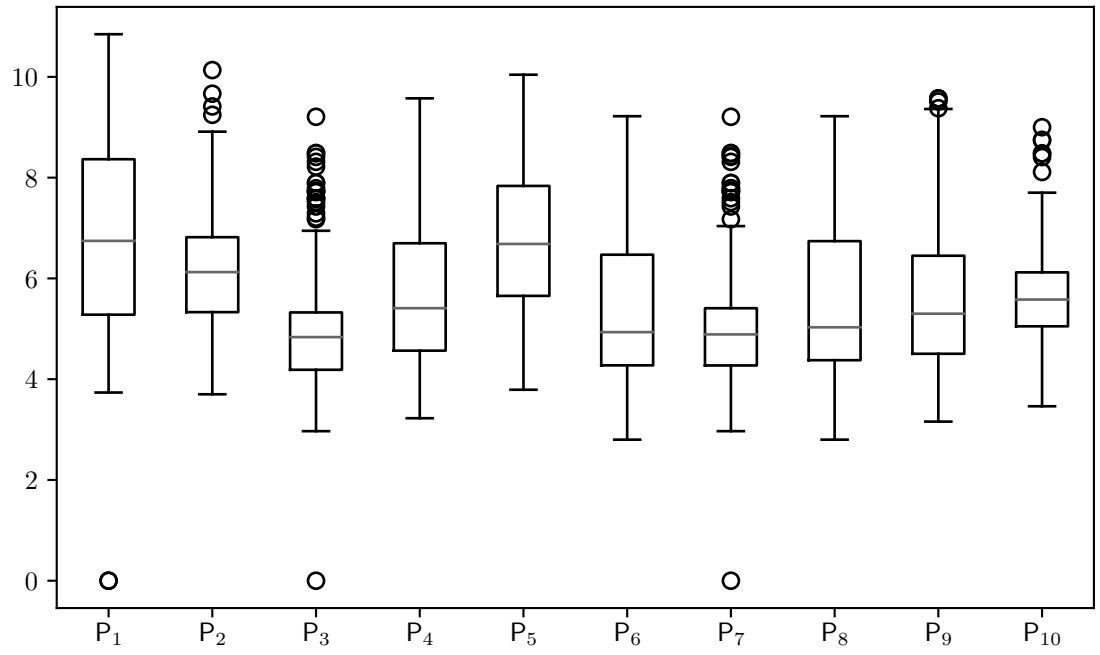
Slika 3.9 pokazuje ovisnost ciljne varijable o tipu promocije. U ovom slučaju vidljiva je statistički bitna razlika između promocije koja uključuje *Dodatno pakiranje* u odnosu na druge promocije. Kod tipa promocije *X% jeftinije* dostižu se najveće vrijednosti. Iz samog dijagrama pravokutnika jasno je da će uključivanje tipa promocije pomoći kod modeliranja ciljne varijable.

Za kontinuirane varijable najlakše je provjeriti korelaciju između njih i ciljne va-



Slika 3.7: Prikazano je ponašanje ciljne varijable u ovisnosti o mediju pomoću kojeg se proizvod oglašava.

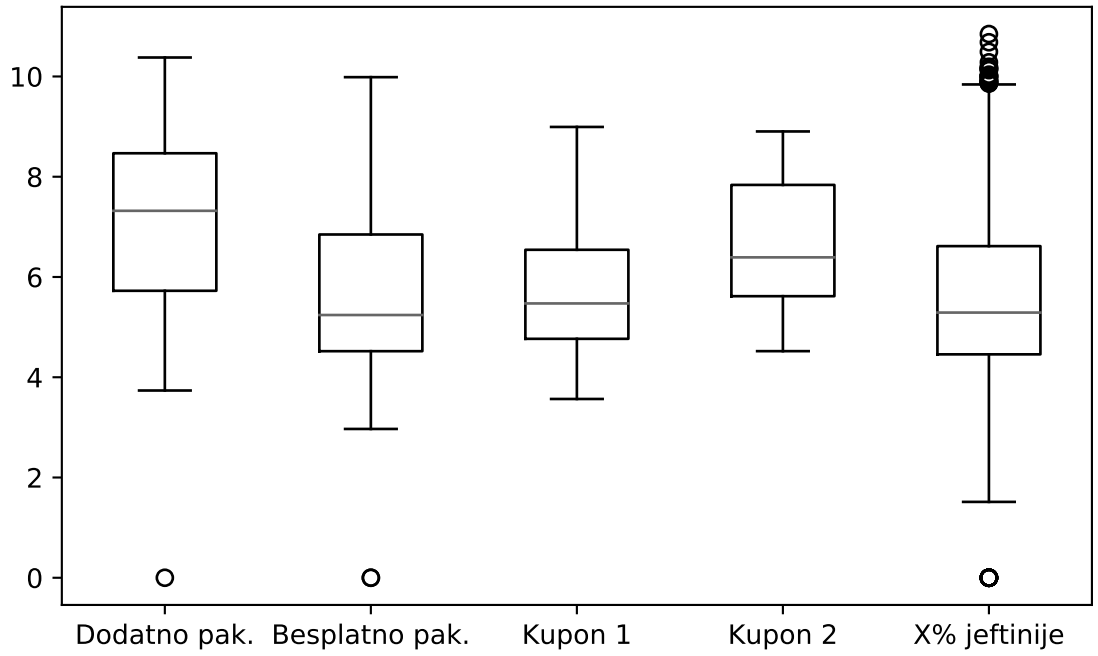
rijable y. U tablici 3.1 dane su vrijednosti korelacija s raznim kontinuiranim varijablama. Između A i B postoji jaka pozitivna korelacija (blizu 1) te su zbog toga vrijednosti korelacije skoro pa identične stoga je moguće eliminirati jednu od njih. Sve varijable su logaritmirane jer se ponašaju po zakonu potencije (slika 3.2). Prikazani su i intervali pouzdanosti, a za njihovu procjenu korišten je postupak *bootstrap*. Distribucije vrijednosti korelacija su Gaussove razdiobe, pa je za izračun intervala pouzdanosti korištena t-distribucija.



Slika 3.8: Prvih deset proizvoda po broju promocija. Prikazan je utjecaj proizvoda na vrijednost ciljne varijable.

Tablica 3.1: Prikazane su pozitivne korelacije s ciljnom varijablom. Varijable A i B predstavljaju procjenu dosega promocije (koliko je letaka podijeljeno, primjeraka novina prodano i dr.).

	$\log y$	95% CI
$\log y$	1.00	—
$\log A$	~ 0.87	(0.8667, 0.8685)
$\log B$	~ 0.86	(0.8561, 0.8579)
Površina ($\log \mathrm{cm}^2$)	~ 0.54	(0.5356, 0.5398)
log-Cijena prije promocije	~ 0.18	(0.1786, 0.1852)
log-Cijena za vrijeme promocije	~ 0.16	(0.1618, 0.1676)



Slika 3.9: Utjecaj tipa promocije na ciljnu varijablu.

Poglavlje 4

Vrednovanje

U ovom poglavlju opisano je vrednovanje modela linearne regresije i aditivnih stabala, implementacija programskog rješenja i opis podataka koji su se koristili za vrednovanje. Većina izvornog koda priložena je u dodatku A i B.

4.1 Implementacija

Korišten je programski jezik PYTHON (verzija 3.6). Za statističku obradu podataka korištene su programske biblioteke SCIPY [4], PANDAS [6] i NUMPY [14]. Za crtanje grafova u ovom radu korištena je biblioteka MATPLOTLIB [3]. Za interaktivnu obradu podataka i razvoj korišten je sustav JUPYTER [12]. Za model linearne regresije korištena je implementacija prisutna u SCIKIT-LEARN [11], a za aditivna stabla implementacija prisutna u LIGHTGBM [7][1].

Obrada podataka

Podaci su iz XLS formata prebačeni u CSV za lakše korištenje. Nakon učitavanja identificirano je nekoliko ekvivalentnih, nepotrebnih i praznih (ili sa samo jednom vrijednošću) stupaca koji su izbačeni. Također su iz podataka izbačeni retci za koje nije bila definirana ciljna varijabla y – ćelija u stupcu koji predstavlja ciljnu varijablu je bila prazna. Tijekom pregleda ručno su označeni stupci koji sadrže kontinuirane i kategoričke vrijednosti.

[1] `https://github.com/Microsoft/LightGBM`

Kodiranje vektora značajki

Nakon obrade potrebno je pretvoriti retke u vektor značajki koji modeli mogu koristiti za učenje. Kontinuirane varijable su direktno preslikane u svoju dimenziju vektora. Za kategoričke varijable korišteno je *1hot* kodiranje (engl. *one-hot encoding*) ilustrirano na slici 4.1. Moglo se koristiti i direktno kodiranje u jednu varijablu. U tom slučaju koristi se preslikavanje poput

$$f(c) = \begin{cases} 1, & c = \mathsf{A2} \\ 2, & c = \mathsf{A3} \\ 3, & \text{inače.} \end{cases}$$

Ovim preslikavanjem uvodi se eksplicitni poredak (što možda nije željeno ponašanje). Kategorija A2 je manja od kategorija A3 i A4. Ako su recimo A2 i A4 korišteni samo za vrijeme blagdana (potrošnja je tada veća), onda linearni model neće moći odijeliti A3 (koji se ne koristi za blagdane) od preostalih (ne može se nacrtati pravac koji dijeli apscisu tako da su 1 i 3 sa suprotne strane od 2). Ovo pogoršava performanse modela jer težina α koju bi dodijelili za blagdanske kategorije ne bi imala dovoljno jak efekt na A2 u linearnom modelu $\alpha f(c) + \beta$.

Kategorija	x_i	x_{i+1}	x_{i+2}
A2	0	0	1
A3	0	1	0
A4	1	0	0

Slika 4.1: Prikazano je *1hot* kodiranje kategorija varijable formata stranice u komponente vektora značajki x.

Nakon provedenog kodiranja za kontinuirane i kategoričke varijable dobivamo vektor značajki x dimenzije $d = 7243$. Skup vektora značajki sadrži $N = 12627$ vektora. Veličine N i d upućuju na to da je možda previše značajki u odnosu na broj vektora stoga je potrebno koristiti regularizaciju tijekom učenja ili izbaciti problematične značajke da bi se smanjila dimenzionalnost (postoje one s nekoliko stotina kategorija). Jedan od mogućih postupaka je zadržati kategorije pojedine varijable iznad nekog broja ponavljanja m, a sve ostale svesti na jednu kategoriju kojom bi označili kategorije s jako malo ponavljanja.

Nakon eliminacije kategorija s više od 20 jedinstvenih vrijednosti (slika 3.1) dimenzija vektora je $d = 116$. Prednost stabla odluke je ta što za kategoričke varijable ne treba vršiti *1hot* kodiranje. Kod izrade stabla lako se operatori $<$ i $>$ zamijene s $=$. Za model koji koristi stablo odluke podaci imaju kompaktniju reprezentaciju

Tablica 4.1: Srednja kvadratna pogreška modela

Model	$\overline{\text{MSE}}$	95% CI
Lin. regresija	~ 0.2083	(0.1950, 0.2216)
DS 5000†	$\mathbf{\sim 0.0357}$	(0.0258, 0.0457)
S 5000†	~ 0.0385	(0.0261, 0.0510)
DS 2000†	~ 0.0374	(0.0273, 0.0476)
S 2000†	~ 0.0394	(0.0270, 0.0518)
S 1000†	~ 0.0418	(0.0293, 0.0543)
S 1000	~ 0.0480	(0.0402, 0.0558)
S 300†	~ 0.0516	(0.0393, 0.0638)
S 300	~ 0.0641	(0.0556, 0.0726)
S 100†	~ 0.0687	(0.0569, 0.0804)
S 100	~ 0.0900	(0.0804, 0.0996)

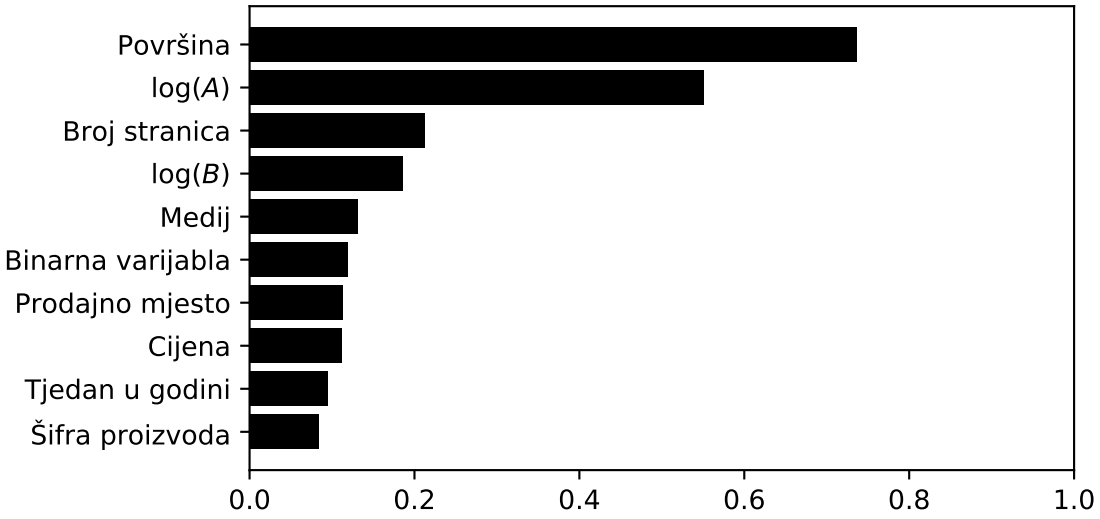
(samo preslikamo sve stupce, kojih je u našem slučaju $k = 56$, u vektor koristeći naivno kodiranje kategoričkih varijabli).

4.2 Rezultati

Rezultati su prikazani u tablici 4.1. Prikazana je srednja kvadratna pogreška (engl. *mean squared error*, MSE). Za svaki model korišten je cijeli podatkovni skup za vrednovanje učinkovitosti. Prikupljanje se vršilo postupkom unakrsne validacije. Skup se rastavio na 10 podskupova jednake veličine i za prikupljanje 10 MSE vrijednosti korišten je svaki pojedinačni podskup za testiranje dok su ostali korišteni za treniranje. Ovo nam omogućuje usporedbu modela.

Modeli S koriste običan algoritam, a DS algoritam DART za učenje stabala. Modeli označeni s † koriste svih $k = 56$ značajki dok ostali za kategoričke značajke koriste samo one kojima je broj jedinstvenih vrijednosti manji od 20. Modeli koji koriste sve značajke su učinkovitiji. Broj koji stoji kraj imena nelinearnog modela predstavlja broj stabla odluke koji zajedno procjenjuju ciljnu varijablu. Što je više stabala, to je MSE manja. Broj čvorova u stablu je fiksiran na 31, ali moglo bi se istražiti postoji li bolji broj. Parametar brzine učenja je $\nu = 0.1$ za stabla S, a $\nu = 1$ za stabla DS. Zbog postojanja $p_{drop} = 0.5$ parametra, tehnika *shrinkage* se ne treba koristiti.

Na slici 4.2 prikazana je važnost varijabli u modelu. Neke su otkrivene kao bitne u poglavlju 3, dok je varijabla koja predstavlja tjedan u godini izgledala nebitno. Šifra proizvoda je očito presudna u slučaju da postoji promocija koja je jako uspješna ili ako se proizvod ponavlja više puta sa sličnim uspjehom.

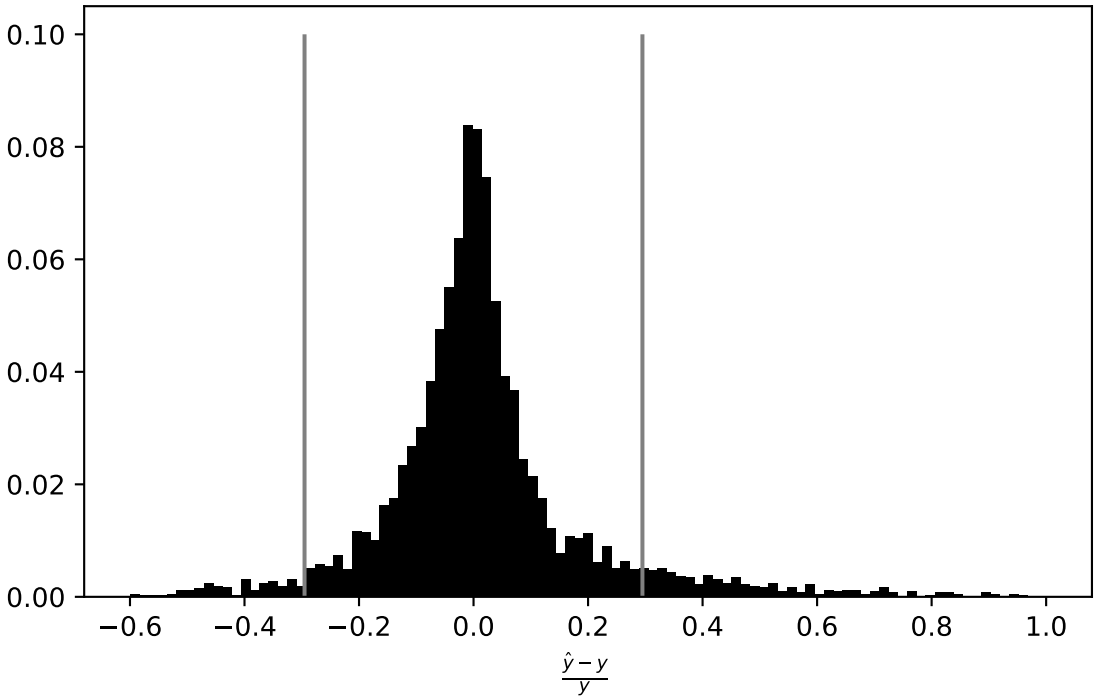


Slika 4.2: Prvih devet varijabli po relativnoj važnosti u modelu

Analiza pouzdanosti predviđanja modela

Očito je iz rezultata da je nelinearni model sa stablima odluke puno učinkovitiji od linearnog modela te može lakše iskoristiti značajke koje bi za linearni model predstavljale preveliko povećanje dimenzionalnosti. Pitanje je kako se optimizacija MSE metrike preslikava u stvarnost. Sigurno bismo željeli ove modele iskoristiti za provjeru uspješnosti buduće promocije. Htjeli bismo znati koju će vrijednost poprimiti ciljna varijabla za određene parametre marketinške promocije, ali bismo istovremeno htjeli znati i kolika je pouzdanost te vrijednosti.

Za vrednovanje korisnosti naučen je najuspješniji model na 8000 primjera, dok su preostali iskorišteni za analizu. Slika 4.3 prikazuje relativnu pogrešku za primjere kod kojih ciljna varijabla poprima vrijednost manju od 3000. Kod predviđanja vrijednosti marketinške promocije može se očekivati pogreška od 2% (srednja vrijednost) mada je raspon na slici puno veći – 90% podataka leži u naznačenom intervalu gdje je apsolutna relativna pogreška oko 30%. Moguće poboljšanje bi bilo priložiti interval pouzdanosti predviđanja ili model učiti koristeći neku drugu metriku umjesto MSE tako da izbjegnemo mogućnost prevelikih pogrešaka. Model bi također mogao odbiti dati predviđanje ako u vrijednost nije dovoljno siguran.



Slika 4.3: Prikaz relativne pogreške za primjere kod kojih ciljna varijabla poprima vrijednost manju od 3000. Vertikalne linije označavaju raspon relativne pogreške na 90% primjera.

Zaključak i otvorena pitanja

Nelinearni modeli korišteni u ovom diplomskom radu su učinkoviti za problem predviđanja ishoda marketinške aktivnosti. U okviru ovog rada nije izvršena pretraga parametara modela (broj čvorova u stablu, broj stabala) koja bi dovela do najbolje učinkovitosti. Za svako predviđanje mogao bi se vratiti i interval pouzdanosti koji daje veću sigurnost u raspon ishoda marketinške aktivnosti. Moguće je i naučiti modele za više zasebnih raspona vrijednosti ciljne varijable koji bi vjerojatno bili učinkovitiji (umjesto pronalaska jednog modela koji treba dobro modelirati cijeli raspon).

Bibliografija

[1] Jerome H Friedman, *Greedy function approximation: a gradient boosting machine*, Annals of statistics (2001), 1189–1232.

[2] Trevor Hastie, Robert Tibshirani i Jerome Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, 2. izd., Springer, 2009.

[3] J. D. Hunter, *Matplotlib: A 2D graphics environment*, Computing In Science & Engineering **9** (2007), br. 3, 90–95.

[4] Eric Jones, Travis Oliphant, Pearu Peterson i ost., *SciPy: Open source scientific tools for Python*, 2001–, `http://www.scipy.org/`, [zadnji pristup 11. studenoga 2017.].

[5] Erich Leo Lehmann i George Casella, *Theory of point estimation*, Springer Science & Business Media, 2006.

[6] Wes McKinney, *Data Structures for Statistical Computing in Python*, Proceedings of the 9th Python in Science Conference (Stéfan van der Walt i Jarrod Millman, ur.), 2010, str. 51 – 56.

[7] Qi Meng, Guolin Ke, Taifeng Wang, Wei Chen, Qiwei Ye, Zhi Ming Ma i Tieyan Liu, *A Communication-Efficient Parallel Algorithm for Decision Tree*, Advances in Neural Information Processing Systems 29 (D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon i R. Garnett, ur.), Curran Associates, Inc., 2016, str. 1279–1287, `http://papers.nips.cc/paper/6381-a-communication-efficient-parallel-algorithm-for-decision-tree.pdf`.

[8] Douglas C Montgomery, Elizabeth A Peck i Geoffrey G Vining, *Introduction to linear regression analysis*, sv. 821, John Wiley & Sons, 2012.

[9] Andrew Ng, *Stanford CS229 - Machine Learning - Ng*, (2008).

[10] Pavle Pandžić i Josip Tambača, *Diferencijalni račun funkcija više varijabli*, 2015, `https://web.math.pmf.unizg.hr/nastava/difraf/predavanja_dir.html`.

[11] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot i E. Duchesnay, *Scikit-learn: Machine Learning in Python*, Journal of Machine Learning Research **12** (2011), 2825–2830.

[12] Fernando Pérez i Brian E. Granger, *IPython: A System for Interactive Scientific Computing*, Computing in Science & Engineering **9** (2007), br. 3, 21–29, `http://aip.scitation.org/doi/abs/10.1109/MCSE.2007.53`.

[13] Robert E Schapire i Yoram Singer, *Improved boosting algorithms using confidence-rated predictions*, Machine learning **37** (1999), br. 3, 297–336.

[14] Stéfan van der Walt, S. Chris Colbert i Gaël Varoquaux, *The NumPy Array: A Structure for Efficient Numerical Computation*, Computing in Science & Engineering **13** (2011), br. 2, 22–30, `http://aip.scitation.org/doi/abs/10.1109/MCSE.2011.37`.

[15] Rashmi Korlakai Vinayak i Ran Gilad-Bachrach, *DART: Dropouts meet multiple additive regression trees*, Artificial Intelligence and Statistics, 2015, str. 489–497.

[16] Xin Yan i Xiao Gang Su, *Linear Regression Analysis: Theory and Computing*, World Scientific Publishing Co., Inc., 2009.

Dodatak A

Učitavanje

Učitavanje podataka izvršeno je u isječku ispod. Izbačeni su nepotrebni stupci i ručno su svrstani u kategoričke i kontinuirane.

```
import csv
import pandas as pd
import numpy as np
# stupci koje izbacujemo jer nam ne trebaju
def find_equivalent_columns(pd_data):
    """Vraća popis stupaca koji su ekvivalentni.
       Tj. imaju identične frekvencije vrijednosti.
    """
    m = dict()
    for col in pd_data:
        counts = tuple(set(pd_data[col].value_counts()))
        m.setdefault(counts, []).append(col)
    return [v for k,v in m.items() if len(v) > 1]
# nakon primjene find_equivalent_columns spremljeni u varijablu
equivalents_to_others = [ ] # definirani stupci (obrisani)
# ne trebaju nam za analizu
unnecessary_columns = [ ] # definirani stupci (obrisani)
columns_to_remove = equivalents_to_others + unnecessary_columns

data = pd.read_csv("/put/do/datoteka.csv",
    encoding='utf-8', # datoteka je enkodirana u utf-8
    sep=';', # polja csv datoteke odvojena su točkom-zarez ';'
    decimal=',', # brojevi su zapisani u decimalnom formatu sa zarezom
)
data = data.dropna(axis=1, # izbaci sve stupce koji su NaN
       how='all', # stupac mora imati sve elemente prazne
      )
# izbacujemo stupce definirane gore
data = data.drop(columns_to_remove, axis=1)
```

```
def find_columns_with_one_value(pd_data):
    """Vraća generator naslova stupaca koji imaju samo jednu vrijednost.
    """
    for col in pd_data:
        vals = pd_data[col].unique()
        if len(vals) == 1:
            yield col

# brišemo stupce koji sadrže samo jednu vrijednost
data = data.drop(find_columns_with_one_value(data), axis=1)
```

Dodatak B

Učenje i vrednovanje

Nakon učitavanja potrebno je očistiti podatke i pretvoriti ih u odgovarajuću reprezentaciju (cijelobrojnu ili broj s decimalnom točkom).

```
categorical, continuous, unknown, goal = feature_types
# prazne elemente popunjava s 'prazno' za kat. var.
def transform_to_categories(pd_data, categorical_headers, to_int=False):
    copy = pd_data.copy()
    for header in categorical_headers:
        copy[header] = copy[header].fillna('prazno')
        if to_int:
            copy[header] = copy[header].astype(np.int64)
        # eksplicitno naznači da su tip kategorije potrebno za stablo
        copy[header] = copy[header].astype('category')
    return copy
# prazne elemente popunjava s 0 za kontinuirane var.
def transform_to_vals(pd_data, continuous_headers):
    copy = pd_data.copy()
    for header in continuous_headers:
        copy[header] = copy[header].fillna(0)
    return copy
# vraća kategorije ispod 20 jedinstvenih vrijednosti
def categories_with_vals_below(dd, below_cnt = 20):
    cats_below = []
    for header in dd.columns:
        if str(dd[header].dtype) != 'category': continue
        cnt = len(dd[header].value_counts())
        if cnt > below_cnt: continue
        cats_below.append(header)
    return cats_below

clean_data = transform_to_categories(data, categorical)
clean_data = transform_to_vals(clean_data, continuous)
```

Ispod je prikazan isječak za vrednovanje najuspješnijeg modela opisanog u ovom radu.

```
import lightgbm.sklearn as lgb
from sklearn.utils import shuffle
# shuffle na podacima za ravnomjeran raspored za unakrsnu validaciju
train_data = shuffle(clean_data, random_state=1331)
# alternativa: samo kategoričke ispod 20 jedinstvenih vrijednosti
# cats = categories_with_vals_below(train_data)
cats = categorical
# definiranje transformacije za primjere X
mapper = DataFrameMapper([
# Za kategoričke vrši se enkodiranje (LabelEncoder) od 1 do N gdje je N
# broj jedinstvenih vrijednosti pojedine kategorije. Stablo zna
# da je to kategorija i neće koristiti operatore < i > za particiju.
    (header, sklearn.preprocessing.LabelEncoder()) for header in cats
# Kontinuirane varijable ispod transformiramo logaritmiranjem.
] + [([header], LogTransformer()) for header in continuous]
, input_df=True, df_out=True) # naznačili smo DataFrame ulaz i izlaz

# definiranje transformacije za ciljnu varijablu y
y_mapper = DataFrameMapper([(['Y'], LogTransformer())], input_df=True)
# primjena transformacije
X = mapper.fit_transform(train_data)
y = y_mapper.fit_transform(train_data).flatten()
# Kategorične varijable eksplictno označene i pretvorene iz broja
# s decimalnom točkom (float) u cijeli broj (int).
X = transform_to_categories(X, cats, to_int=True)

reg = lgb.LGBMRegressor(objective='l2', n_estimators=5000,
                        num_leaves=31, boosting_type='dart')
scores = cross_val_score(reg, X, y, cv=10,
                         scoring=make_scorer(mean_squared_error))
print("MSE: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2))
print(scores)
```

Varijabla scores koristila se za izračun intervala pouzdanosti.

Za učenje modela linearne regresije korišten je isječak ispod.

```
# Npr. varijabla Zemlja1 čije su kategoričke vrijednosti [USA, UK, JP],
# LabelBinarizer će napraviti vektore značajki redom
# [0, 0, 1] - JP, [1, 0, 0] - USA i [0, 1, 0] - UK.
# Ako postoji još jedna kategorička varijabla Zemlja2,
# onda će [1, 0, 0, 0, 0, 1] označavati vektor USA-JP.
from sklearn.utils import shuffle
train_data = shuffle(clean_data, random_state=1331)

mapper = DataFrameMapper([
    (header, LabelBinarizer())
    for header in categories_with_vals_below(train_data)
    # kategoričke varijable s manje od 20 jedinstvenih vrijednosti
] + [([header], LogTransformer()) for header in continuous]
, input_df=True)
# ovo je potrebno ako se koriste sve kategoričke varijable
# sa sparse se naznačava da 0 ne postoji u mem. prikazu podataka
# ], sparse=True, input_df=True)

y_mapper = DataFrameMapper([(['Y'], LogTransformer())], input_df=True)

X = mapper.fit_transform(train_data) # mapiramo podatke u vektore značajki
X = X.astype(np.float64) # najefikasnije za sklearn modele
y = y_mapper.fit_transform(train_data)

print('Podaci: ~{:.2f} MB'.format(X.data.nbytes/1024/1024))
print('Ciljna varijabla: ~{:.2} MB'.format(y.nbytes/1024/1024))
# da se ne stavi sparse veličina cijelih podataka bila bi oko 600MB

# Cilj je izbjeći veliku dimenzionalnost eliminacijom varijabli
# koje imaju puno kategoričkih vrijednosti. Npr. neke imaju
# preko 300 različitih vrijednosti. Povećanje dimenzionalnosti
# bi bilo preveliko za ovako jednostavan model. Jednostavno
# rješenje uzelo bi najmanje učestale vrijednosti pojedinačne
# kategorije i stavilo u jednu vrijednost 'Ostale'.
from sklearn import linear_model
from sklearn.model_selection import cross_val_score
from sklearn.metrics import mean_squared_error, make_scorer

reg = linear_model.Ridge(alpha=0.1, solver='lsqr')
scores = cross_val_score(reg, X, y, cv=10,
                         scoring=make_scorer(mean_squared_error))
print("MSE: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2))
print(scores)
```

Izvorni kod za crtanje slike 4.3.

```
def eval_err(s_data, regressor, th=3000):
    d = s_data
    d = d[d['Y'] < th]
    X = mapper.transform(d)
    X = transform_to_categories(X, cats, to_int=True)
    y = d['Y'].as_matrix()
    yc = np.expm1(regressor.predict(X))
    diff = yc - y
    plt.figure()
    perc_dev = diff / (y + 1)
    l = np.max(perc_dev)
    u = np.min(perc_dev)
    for i, (_, v) in enumerate(sorted(
                        zip(list(np.abs(0-perc_dev)),
                            list(perc_dev)))):
        if i/len(perc_dev) > .9:
            break
        l = min(l, v)
        u = max(u, v)
    plt.xlabel('\\frac{yc-y}{y}')
    weights = np.ones_like(perc_dev)/len(perc_dev)
    plt.hist(perc_dev, weights=weights, bins=np.linspace(-.6, 1, 100))
    plt.vlines( x=l, ymin=0, ymax=.1, color='gray', label='95% CI' )
    plt.vlines( x=u, ymin=0, ymax=.1, color='gray' )
    savefig('relerr'+str(th))
    return np.median(perc_dev), np.mean(perc_dev), l, u
```

Dodatak C

Opisna statistika

Isječak koda ispod je korišten za analizu pojedinačnih kategorija neke kategoričke varijable i njihovog utjecaja na ciljnu varijablu. Željelo se zaključiti je li npr. papir određene veličine poprima veću vrijednost ciljne varijable y od druge. Pomoću funkcije ispod, koristeći neparametarski statistički test može se saznati za sve parove pojedinačnih kategorija jedne varijable vrijedi li nulta hipoteza ili ćemo ju odbaciti i prihvatiti alternativu (da je za kategoriju x_1 ciljna varijabla veća od one za kategoriju x_2).

```
import scipy.stats as sts
import numpy as np
import itertools as it
def greater_test(col_name):
    m = {}
    for col, d in list(data[[col_name, 'Y']].groupby(by=col_name)):
        if len(d) < 10:
            continue
        m[col] = d
    res = {}
    for val1, val2 in it.combinations(m.keys(), 2):
        x1, x2 = np.log1p(m[val1]['Y']), np.log1p(m[val2]['Y'])
        st, p = sts.mannwhitneyu(x1, x2, alternative='greater')
        res[(val1, val2)] = p
    return res
```

Funkcija ispod korištena je za kreiranje dijagrama pravokutnika u opisnoj statistici. Nekad je potrebno rotirati tekst da bi stao na os.

```
def cat_y_boxplot(cat_data, cat='Year', file_name=None,
                  fs=figsize(1), cutoff=50, rotate=90, remap={}):
    cat_y = cat_data[[cat, 'Y']]
    poss = []
    xaxis = []
```

```
    grps = []
    posi = 1
    for pos, grp in sorted(cat_y.groupby(by=cat)):
        if len(grp) < cutoff:
            continue
        poss.append(posi)
        posi += 1
        xaxis_label = pos if type(pos) != np.int64 else pos
        xaxis_label = remap.get(xaxis_label, xaxis_label)
        xaxis.append(xaxis_label)
        grps.append(np.log1p(grp['Y']))
    m=0.9
    plt.figure(figsize=fs)
    plt.boxplot(grps, positions=poss)
    rot = rotate
    plt.xticks(poss, xaxis, rotation=rot)
    plt.tight_layout()
    savefig(file_name)

# boxplot po godini
cat_y_boxplot(data, 'Year', file_name='godina_y', rotate=0)
# boxplot po trajanju promocije u danima
cat_y_boxplot(data, 'PromDays', file_name='promdani_y',
                fs=figsize(2), rotate=0)
```

Izračun intervala pouzdanosti za korelaciju prikazanu u tablici 3.1.

```
import itertools as it
import scipy.stats as sts

cor_data = data.copy()
cor_data['Y'] = np.log1p(cor_data['Y'])
for v in ['Discount', 'log A', 'log B', 'Površina',
          'PBP (Price Before Promotion)', 'PP (Promotion Price)']:
    corrs = []
    # logaritmiranje svega osim popusta
    cor_data[v] = np.log1p(cor_data[v]) if v != 'Discount' else cor_data[v]
    for i in range(5000):
        k = ['Y', v]
        # uzorkovanje i računanje korelacije
        d = cor_data[k].sample(n=100, replace=False).corr().loc['Y', v]
        corrs.append(d)
    # ispis intervala pouzdanosti
    print(v, np.mean(corrs),
          sts.t.interval(0.95, len(corrs)-1,
                         loc=np.mean(corrs), scale=sts.sem(corrs)))
```

Sažetak

Marketinške aktivnosti u maloprodaji bitna su komponenta i veći dio troška marketinške strategije poduzeća. Vrlo je korisno imati sustave koji bi mogli pomoći pri procjeni učinkovitosti specifične marketinške aktivnosti. U ovom radu ostvaren je model za predviđanje ishoda (zarade) uzevši u obzir karakteristike marketinške aktivnosti. Dan je osvrt na dvije metode iz područja matematičke statistike i strojnog učenja. Opisan je model linearne regresije i aditivnih stabala te je za svaki opisana metoda učenja. Dana je i usporedba između više inačica ostvarenih modela te je naposljetku vrednovana korisnost najbolje inačice.

Summary

Marketing activities in retail are an important component and major cost in marketing strategy of a company. It is very useful to have systems which are capable of estimating the outcome of a specific marketing activity. This thesis describes the model for predicting the outcome (yield) of a marketing activity by taking all of its characteristics into account. Two methods from the field of mathematical statistics and machine learning are described. Linear regression and additive trees models are covered and for each a description of learning methods is given. Comparison between several implemented models is given and evaluation of usefulness is done for the best model.

Životopis

Rođena sam 13. rujna 1992. godine u Splitu, gdje sam završila osnovnu školu i Prvu jezičnu gimnaziju. 2012. godine upisujem preddiplomski sveučilišni studij Matematika na Prirodoslovno-matematičkom fakultetu Sveučilišta u Zagrebu, koji završavam 2015. godine stekavši naziv sveučilišna prvostupnica matematike. Iste godine upisujem diplomski studij Financijske i poslovne matematike koji završavam ovim radom.